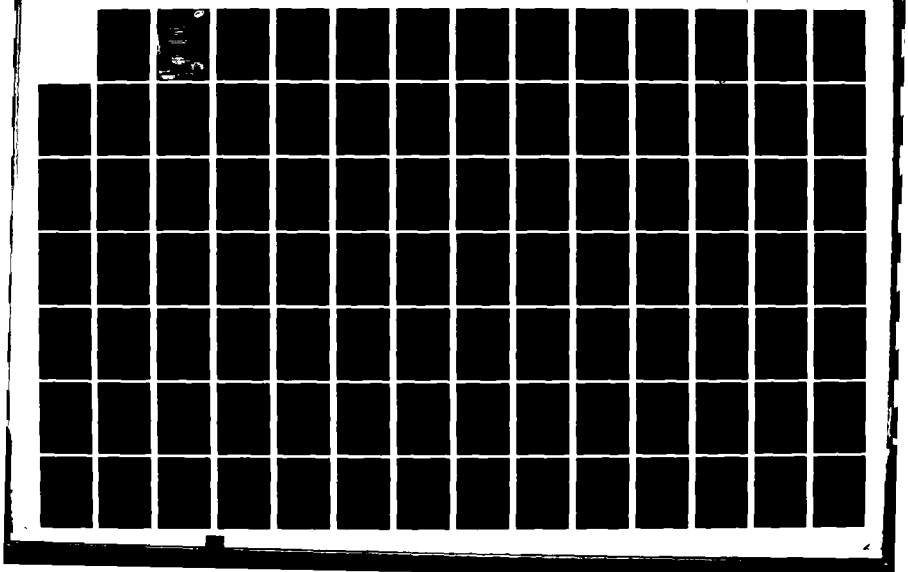
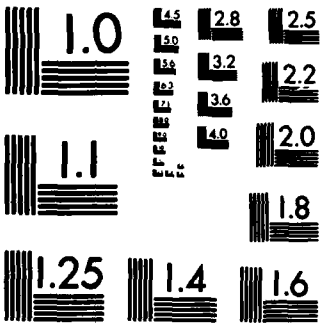
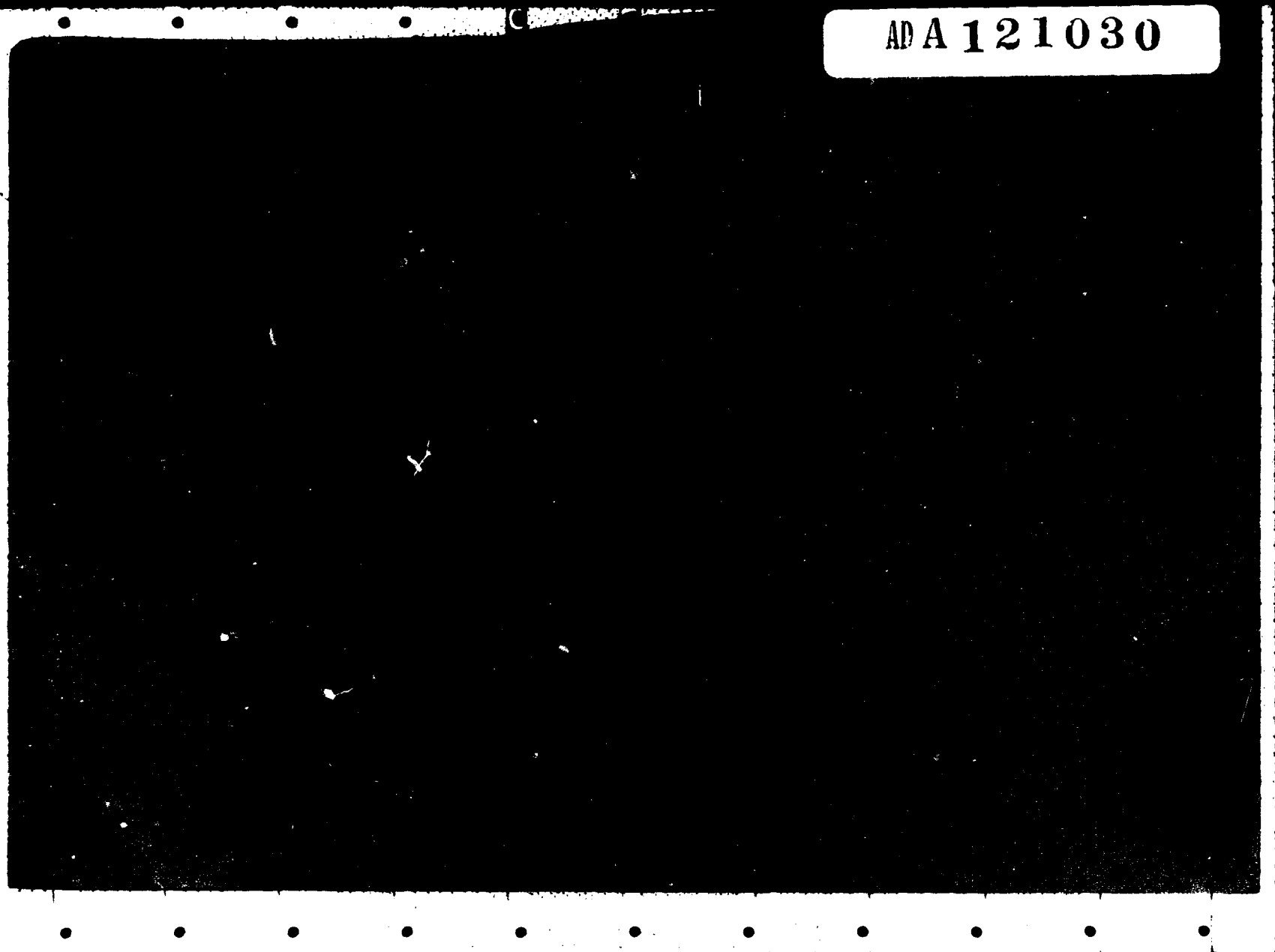
AD-A121 030 DEVELOPMENT OF A WAR READINESS SPARES KIT MODEL AND SOLUTION TECHNIQUE(U) OKLAHOMA STATE UNIV STILLWATER SCHOOL OF INDUSTRIAL ENGINEERI.. M P TERRELL ET AL. 1/2

UNCLASSIFIED 31 DEC 80 F33600-80-C-0423 F/G 9/2 NL







PREFACE

This final report describes work completed under Research Number F33600-80-C-0423 during the period July 1980 through December 1980. This effort was performed for the United States Air Force Logistics Command at Wright-Patterson Air Force Base, Ohio.

The work was performed by the School of Industrial Engineering and Management at Oklahoma State University. Dr. M. Palmer Terrell and Dr. Philip M. Wolfe were co-principle researchers, assisted by Ph.D. students, Mr. Umit Yuceer and Mr. Shawn Yu.

This effort was undertaken to develop a new model and approach for solving the War Readiness Spares Kit (WRSK) problem. The development and evaluation of a new model and computer program, referred to as the Greedy Algorithm, is described herein.

The final report is divided into six chapters and four appendices.

TABLE OF CONTENTS

Page No.

LIST OF TABLES

LIST OF FIGURES

PART I

INTRODUCTION

Problem Statement

The management of spare parts inventories is indispensable to the logistics operations of the Air Force. Effective inventory management systems are of great importance to the Air Force since a large amount of money is spent annually on purchases of spare parts. Two systems 'METRIC and MOD-METRIC' have been implemented to assist in the management of recoverable spare parts. However, some of the logistics problems need specialized and sophisticated solution strategies and methods.

A special type of inventory is the war readiness spares kit (WRSK). This consists of a group of spare parts that are deployed with a flying unit. A WRSK system is in operation and assists in determining the spare parts that should be in a kit. The objective of this system is to determine the contents of a kit which will result in a high level of performance at a minimum cost.

An important aspect of this problem is the definition of measures of supply performance which are operationally meaningful. A thorough investigation of the problem and the underlining assumptions suggests that the NORS (Not Operationally Ready-Supply) and SDO (Stock Due Out) measures can be used to describe the performance of the kit. (See references [3] and [6].) The NORS function is non-linear and nonseparable over the nonnegative integers. This makes the problem difficult to solve. In addition, the number of items in the kit is very large in practice. Consequently, a large size non-linear integer programming needs to be solved. Dynamic programming [1] or the branch-

and-bound method [5] could be employed to solve the WRSK problem, but the computation time required is impractical.

Brief Description of Current DO29 WRSK Model

The current DO29 Model and its computer program is designed to solve this problem. It has two passes. The first pass tries to determine a kit which will yield a high performance for the initial six day period for a squadron of aircrafts. The second pass attempts to find the contents of the whole kit for the thirty day period. A detailed description is provided by Wright Patterson Air Force Base Logistics Command in [8].

This model employs marginal analysis to obtain a kit which will perform better than the requirement kit 57-18. The tradeoff between NORS and SDO is handled by manipulating the weight between them at each step according to some heuristic rules. In addition, at each step of the algorithm some number of units of an item are added to the kit. This number is also manipulated according to some heuristics. This is necessary because the current DO29 Model uses marginal improvements which must be recalculated after one unit of an item is added to the kit due to the nonseparability of the expected NORS function, E(NORS/X).

The current DO29 computer program has some special subroutines to handle input, word packing and unpacking, and calculation and interpolation of probabilities. One of the subroutines provides intermediate kits at various budget levels.

Problems With Current Solution Methodology

The marginal analysis as currently used will not guarantee an optimum solution. Fox [4] states conditions when discrete marginal analysis will

result in an optimum answer. One criterion is that the functions must be separable. The model used in the WRSK system involves functions which are not separable. Consequently, the marginal analyis approach as applied to the WRSK must be categorized as a heuristic.

Another problem involves the calculation of E(NORS/X) and E(SDO/X) functions. These expectation formulas are truncated to ease the computations. Truncation may cause numerical errors if the upper bound of the truncation is not chosen suitably.

Relative to the current WRSK system, models involving large flying units (greater than 24 aircraft) require a large amount of memory. So much so, that portions of the resident software must be removed to provide the required storage. In order to avoid the use of excessive storage, word packing and unpacking is employed. This however creates some numerical errors throughout the computations. Some of the probability values are stored and others are interpolated when needed. This causes some probabilities to be underestimated. More importantly, this process is very time consuming. Approximately half of the execution time is spent on packing and unpacking.

PART II

OBJECTIVES OF THIS RESEARCH

General

The objective of this research has been to develop, validate, and document an improved War Readiness Spares Kit (WRSK) model and solution procedure which may be used by the United States Air Force in the management of recoverable spare parts.

This research effort has been achieved by accomplishing a series of subobjectives. These objectives are stated below, and reference is made to the pertinent parts of this report.

Objectives

Objective 1: Develop and validate an improved solution procedure for the current WRSK model.

Development and validation of such a solution method was accomplished in two stages. In the first stage, measures of supply performance and their mathematical properties were investigated. Part III of this report is devoted to the results of this investigation. The second stage was the development of a mathematical model which is reported in Part IV where a multiobjective approach is proposed for the WRSK problem. Potential solution methods were then investigated. A small example problem was solved by these algorithms in order to demonstrate how they generate solutions and how they converge to an optimum. Evaluation of these methods led to the selection of a preferred solution method which was investigated and tested using F4D data. The final section of Part IV discusses the current D029 model which has features pertinent to this research.

Objective 2: Develop procedures which will eliminate the problems caused by using expected value approximations.

Part III considers the measures of supply performance which are closely related to the stocastic behavior of a WRSK kit. Accurate expected value computations are of vital importance to the generation of high quality solutions. Truncation of an infinite sum is compared to the approximation of the sum. Proper use of the truncation procedure is explained by some examples in the last section of Part III.

Objective 3: Implement the algorithm to solve the War Readiness Spares Kit problem.

A major aspect of this research has been to implement the proposed solution method and use it to solve a WRSK problem with real data. The implementation and computational features using F4D data is discussed in Part V.

Objective 4: Compare the performance of the algorithm with the current system in terms of time and computational feasibility.

In order to establish a base for meaningful comparison, the current D029 computer program was modified for compatibility with IBM compilers. Then, performance of the two algorithms was observed in terms of time and computational feasibility. These results are discussed in Part V.

Objective 5: Document results and computer programs.

This report, in its totality, documents the research results. The detailed documentation and FORTRAN listing of the proposed solution method is given in Appendix B.1 and B.2. In addition, the FORTRAN listing of the IBM modified D029 program is provided in Appendix C.1 and C.2.

PART III

MEASURES OF SUPPLY PERFORMANCE

Introduction

This section focuses on measures of supply performance. The stochastic prediction of the behavior of a kit during the mission is of vital importance in the study of the WRSK problem. Another important aspect, when measuring the performance of a kit, is the relationship of the performance measures to the operations.

The performance of a kit can be measured in several ways; fill rate, operational rate, average back orders, average number of Not Operationally Ready Supply Aircraft, and kit cost. These can be described briefly as follows. Fill rate is the ratio of the number of parts issued over a fixed time period to the number of parts demanded over the same period. Operational rate is the probability that there will be no due outs from the base supply at any time. Back orders contain the number of stock due-outs (SDO) from base supply during a fixed period. Finally, NORS is the number of grounded aircraft due to lack of spare parts. Experience has revealed that average SDO, average NORS, and the cost of a kit relate to operations better than the other measures.

Mathematical Development of Performance Measures

The following definitions and assumptions are made in order to develop the mathematical expressions for average back orders and NORS.

(1) X represents a kit consisting of individual items i of quantity x_i.

(2) n items are under consideration for the kit.

(3) The demand of an item is a Poisson distributed random variable.

(4) The failure of any item is independent of the failure of any other item.

(5) When a demand for an item can not be satisfied from the stock on hand, the required part is consolidated by cannibilization.

Let $F_i(\cdot)$ denote the cumulative Poisson sums for the item i. The operational rate given no more than k planes for cannibilization is expressed by the statement

$$b_k = P[NORS \leq k/X] = \prod_{i=1}^{n} F_i(x_i + ka_i), \quad k = 0,1,2,\ldots,N$$

where a_i is the number of applications of the part i on an airplane, and N is the total number of aircraft in the flying unit. Consequently

$$\Delta b_k = P[\#\text{ of NORS} = k] \quad \text{for } k = 0,1,2,\ldots,N$$

then $$\Delta b_0 = P[\#\text{ of NORS} = 0] = b_0$$

and $$\Delta b = P[\#\text{ of NORS} \leq k] - P[\#\text{ of NORS} \leq k-1]$$

$$= b_k - b_{k-1} \quad \text{for } k = 1,2,3,\ldots,N$$

Finally $$E(NORS/X) = \sum_{k=1}^{N} (b_k - b_{k-1})$$

An equivalent development of the expectation of the NORS function for a given kit X goes as follows.

$$E(NORS/X) = \sum_{k=0}^{\infty} k\, P[\#\text{ of NORS} = k]$$

$$E(NORS/X) = \sum_{k=0}^{\infty} (1 - P[\#\text{ of NORS} \leq k])$$

$$E(NORS/X) = \sum_{k=0}^{\infty} \left(1 - \prod_{i=1}^{n} F_i(x_i + ka_i)\right)$$

An approximation to the above function for computational purposes was proposed by B. L. Miller in (6) as follows.

$$E(NORS/X) \simeq \sum_{k=0}^{m} \left(1 - \prod_{i=1}^{n} F_i(x_i + ka_i)\right) + \sum_{k=m+1}^{\infty} \sum_{i=1}^{n} 1 - F_i(x_i + ka_i)$$

where m is such that $\prod_{i=1}^{n} F_i(x_i + ka_i)$ is very close to 1 for all k>m. See Figure 3.1.

The expected back orders, SDO, for a given kit X is obtained by

$$E(SDO/X) = \sum_{i=1}^{n} \sum_{j=o}^{\infty} (1 - F_i(x_i + j))$$

Reference may be made to [3] and [8] for further details.

If item i costs \$ c_i, the cost of the kit is then $\sum_{i=1}^{n} c_i x_i$ or in vector form, CX.

Mathematical Properties of Performance Measures

A few properties of the measures, E(NORS/X) and E(SDO/X), are presented below. They will be used later in the development of solution methods and algorithms.

Property 1

Both E(SDO/X) and E(NORS/X) are monotonically decreasing because the first differences are nonpositive.

Define $\Delta_k E(SDO/X) = E(SDO/X + e_k) - E(SDO/X)$ where e_k represents the kth unit vector. Then

$$\Delta_k E(SDO/X) = \sum_{\substack{i=1 \\ i\neq k}}^{n} \sum_{j=0}^{\infty} [1 + F_i(x_i + j)] + \sum_{j=0}^{\infty} [1 + F_k(x_k + 1 + j)] - \sum_{i=1}^{n} \sum_{j=0}^{\infty} [1 - F_i(x_i + j)]$$

$$= \sum_{j=1}^{\infty} [1 - F_k(x_k + j)] - \sum_{j=o}^{\infty} [1 - F_i(x_k + j)] \quad \text{which yields}$$

$\Delta_k E(SDO/X) = -[1 - F_k(x_k)]$, the marginal improvement in average SDO

when a one unit of item k is added to the kit.

Likewise, the marginal improvement in average NORS is calculated as follows.

Define $\Delta_k E(NORS/X) = E(NORS/X + e_k) - E(NORS/X)$. Then

$$\Delta_k E(NORS/X) = \sum_{j=o}^{\infty} [1 - \prod_{\substack{i=1 \\ i=k}}^{n} F_i(x_i + ja_i) F_k(d_k + ja_k + 1)] - \sum_{j-o}^{\infty} [1 - \prod_{j=1}^{n} F_i(x_i + ja_i)]$$

$$= \sum_{j=o}^{\infty} \left(\left[\prod_{\substack{i-1 \\ i \neq k}}^{n} F_i(x_i + ja_i) \right] [F_k(x_k + ja_k) - F_k(x_k + ja_i + 1)] \right)$$

$$= - \sum_{j=o}^{\infty} P_k (x_k + ja_k + 1) \prod_{\substack{i=1 \\ i \neq k}}^{n} F_i (x_i + ja_i)$$

where $P_k(s) = \dfrac{e^{-\mu_k} \mu_k^s}{s!}$

Property 2

$E(SDO/X) \geq E(NORS/X)$ for any X

Proof:

$$E(SDO/D) = \sum_{j=1}^{m} \sum_{i=1}^{n} [1 - F_i(x_i + j)] + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} [1 - F_i(x_i + j)]$$

$$E(NORS/X) = \sum_{j=o}^{\infty} [1 - \prod_{i=1}^{n} F_i(x_i + ja_i)] + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} [1 - F_i(x_i + ja_i)]$$

Define DIF = E(SDQ/X) - E(NORS/X)

$$= \sum_{j=o}^{m} \left\{ \sum_{i=1}^{n} [1 - F_i(x_i + j)] - 1 + \prod_{i=1}^{n} F_i(x_i + ja_i) \right\} + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} [F_i(x_i + ja_i) - F_i(x_i + j)]$$

$$= \sum_{j=o}^{m} \left\{ n - 1 - \sum_{j=1}^{n} F_i(x_i + j) + \prod_{i=1}^{n} F_i(x_i + ja_i) \right\} + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} [F_i(x_i + ja_i) - F_i(x_i + j)]$$

Since $a_i \geq 1$ for all i, then $1 \geq F_i(x_i + ja_i) \geq F_i(x_i + j)$ and

$$\prod_{i=1}^{n} F_i(x_i + ja_i) \geq \prod_{i=1}^{n} F_i(x_i + j), \text{ further}$$

$$\sum_{i=1}^{n} F_i(x_i + j) - \prod_{i=1}^{n} F_i(x_i + ja_i) \leq \sum_{i=1}^{n} F_i(x_i + j) - \prod_{i=1}^{n} F_i(x_i + j).$$

Let $\Psi(Y) = \sum_{i=1}^{n} y_i - \prod_{i=1}^{n} y_i$. The maximum of this function is (n-1). The proof goes as follows:

$$\nabla\Psi = (1 - \prod_{i=2}^{n} y_i, \ldots, 1 - \prod_{i\neq k}^{n} y_i, \ldots, 1 - \prod_{i=1}^{n-1} y_i)$$

$$\nabla^2\Psi = \begin{bmatrix} o & a_{12} & \cdots & a_{1n} \\ a_{21} & o & \cdots & a_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \qquad \text{where } a_{kj} = -\prod_{\substack{i\neq k \\ i\neq j}} y_i$$

$$\text{and} \quad |a_{kj}| \leq 1$$

Clearly $r^t \nabla^2\Psi r < 0 \quad r\epsilon R^n_+$. Hence Ψ is concave in the positive unit hypercube. Furthermore, $\nabla\Psi(y) = 0$ yields the solution that $y_i^* = 1$ for all i=1,...,n. Consequently $\Psi(\tilde{1}) = n-1$ is the maximum of the function.

Finally DIF ≥ 0, since $(n-1) - [\sum_{i=1}^{n} F_i(x_i + j) - \prod_{i=1}^{n} F_i(x_i + ja_i)] \geq 0$ and

$$\sum_{i=1}^{n} [F_i(x_i + ja_i) - F_i(x_i + j)] \geq 0 \quad \text{for all } j \geq 0.$$

Property 3

The E(NORS/X) function can be approximated by a separable function.

This is accomplished as follows (also see [3]).

$$E(NORS/X) \simeq (m+1) + \sum_{i=1}^{n} \sum_{k=o}^{m} b_k \log F_i(x_i + ka_i) - \sum_{k=m+1}^{\infty} [1 - F_i(x_i + ka_i)]$$

where $b_k = \prod_{i=1}^{n} F_i(\bar{x}_i + ka_i)$ for some $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots \bar{x}_n)$.

where n is such that $\prod_{i=1}^{n} F_i(x_i + ka_i)$ is very close to 1 for all k>m.

The term $\sum_{k=o}^{m} [1 - \prod_{i=1}^{n} F_i(x_i + ka_i)]$ can be rewritten as $(m+1) - \sum_{k=o}^{m} \prod_{i=1}^{n} F_i(x_i + ka_i)$.

The product $\prod_{i=1}^{n} F_i(x_i + ka_i)$ can be replaced by some multiple of its logarithm in the following manner.

$-b \sum_{i=1}^{n} \log F_i(x_i + ka_i)$ where b_k is a parameter defined by $b_k = \prod_{i=1}^{n} F_i(\bar{x}_i + ka_i)$ for some $\bar{X} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$.

Finally then, separability is accomplished as E(NORS/X) = (m+1) +

$$\sum_{i=1}^{n} \sum_{k=o}^{m} b_k \log F_i(x_i + ka_i) + \sum_{k=m+1}^{\infty} [1 - F_i(x_i + ka_i)].$$

This results plays an important role in the development of an algorithm which is discussed in Section IV.

Expected Value Calculations

Calculation of the expected NORS function for a given kit X; $E(NORS/X) = \sum_{j=1}^{\infty} (1 - \prod_{i=1}^{n} F_i(x_i + ja_i))$ creates computational difficulties. This expectation is an infinite sum. Since it is convergent, the terms become negligibly small for sufficiently large j's. There are two ways to handle the computation. One way is to truncate this infinite sum for a sufficiently large value M. The second way is to develop an approximation as suggested by B. L. Miller in [6]. Physically the number NORS airplanes can not exceed the number of airplanes in the squadron. This then establishes a natural upper bound for truncation Both truncated and approximate expected NORS formulas are provided below, and the difference between them is evaluated.

a) Truncation: $f_T(X) = \sum_{j=0}^{M} (1 - \prod_{i=1}^{n} F_i(x_i + ja_i))$

b) Approximation: $f_A(X) = \sum_{j=0}^{m} (1 - \prod_{i=1}^{n} F_i(x_i + ja_i)) + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i))$

where M is the number of planes in the squadron, a_i is the number of applications of the part i on an airplane, and m is such that $\prod_{i=1}^{n} F_i(x_i + ja_i)$ is very close to 1 for all $j>m$.

The difference is given by the relation $\Delta = f_A(X) - f_T(X)$. If $m > M$, then

$$\Delta = \sum_{j=M+1}^{m} (1 - \prod_{i=1}^{n} F_i(x_i + ja_i)) + \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i)).$$

If $m \leq M$, then $\Delta = \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i)) - \sum_{j=m+1}^{M} (1 - \prod_{i=1}^{n} F_i(x_i + ja_i))$

$$\text{or } \Delta = \sum_{j=m+1}^{\infty} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i)) - \sum_{j=m+1}^{M} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i))$$

because the approximation is valid for all $j>m$. Finally

$$\Delta = \sum_{j=M+1}^{\infty} \sum_{i=1}^{n} (1 - F_i(x_i + ja_i))$$

In either case, Δ is a positive term and Δ may not be negligibly small especially for large squadrons with some items having heavy demand rates.

The following examples illustrate the magnitude of the difference between the truncated and approximated formulas.

Example 1

Suppose that there are three planes in a squadron, and there are two essential items A, B with demand rates $\mu_A = 5$ and $\mu_B = 5.5$. The initial stock levels are determined to be $x_A = 5$ and $x_B = 6$. For simplicity, $a_A = a_B = 1$.

a) Calculation of NORS by $f_T(X)$

j	$F_A(5+j)$	$F_B(6+j)$	Product	1 - Product
0	.616	.686	.42258	.57742
1	.762	.809	.61646	.38354
2	.867	.894	.77510	.22490
				$f_T(x) = 1.18586$

b) Calculation of NORS by $f_A(X)$

m is chosen as 6

j	$F_A(5+j)$	$F_B(6+j)$	Product	1 - Product
0	.616	.686	.42258	.57742
1	.762	.809	.61646	.38354
2	.867	.894	.77510	.22490
3	.932	.946	.88167	.11833
4	.968	.975	.94380	.05620
5	.986	.989	.97515	.02485
6	.995	.996	.99102	.00898
				Sum 1.39422

j	$1 - F_A(5+j)$	$1 - F_B(6+j)$
7	.00202	.00160
8	.00070	.00060
9	.00022	.00020
10	.00007	.00006
11	.00002	.00001
	Sum .00303	.00247 TSUM = .00550

$f_A(X) = 1.39972$

$\Delta = 0.21385$

Example 2

Suppose there are three items A, B and C with $\mu_c = 4.5$ and $x_c = 5$

$m = 6$ again.

a) Calculation by truncated formula

j	$F_A(5+j)$	$F_B(6+j)$	$F_C(5+j)$	Product	1 - Product
0	.616	.686	.720	.30426	.69574
1	.762	.809	.884	.52029	.47971
2	.867	.894	.921	.71387	.28613
					$f_T(X) = 1.46158$

b) Calculation by Approximated formula $f_A(X)$

j	$F_A(5+j)$	$F_B(6+j)$	$F_C(5+j)$	Product	1 - Product
0	.616	.686	.720	.30426	.69574
1	.762	.809	.844	.52029	.47971
2	.867	.894	.921	.71387	.28613
3	.932	.946	.964	.84993	.15007
4	.968	.975	.985	.92964	.07036
5	.986	.989	.994	.96930	.03070
6	.995	.996	.998	.98904	.01096
					Sum = 1.72367

i	$1 - F_A(5+j)$	$1 - F_B(6+j)$	$1 - F_C(5+j)$	
7	.00202	.00169	.00066	
8	.00070	.00060	.00020	
9	.00022	.00020	.00006	
10	.00007	.00006	.00002	
11	.00002	.00001	---	
			.00094	TSUM = .00644

$$f_A(X) = 1.73011$$

$$\Delta = .26853$$

These small examples show that there is an error which is not negligible. Even if the number of airplanes in the squadron is sufficiently large, one has to be careful to reduce the magnitude of this error. In addition, there will be round-off errors for large size problems even with double-precision arithmetic. It is essential to know the origin of the errors if one wishes to reduce the magnitude of the errors.

PART IV

MATHEMATICAL MODELLING AND SOLUTION METHODS FOR THE WRSK PROBLEM

A Multiobjective Approach to the WRSK Problem

In simple terms, the objective of the WRSK problem is to determine the quantity of each part in the kit which will minimize the average NORS, average SDO, and the cost of the kit simultaneously. All of these three objectives conflict with each other. If cost is minimized, then the other two measures increase. If average NORS is minimized for a fixed cost, the average SDO turns out very high. Conversely, the average NORS comes out high when the average SDO is minimized for a fixed budget level. The relationship between average NORS, SDO, and the cost was discussed in [3] and [8]. The WRSK problem is a multiobjective programming problem, and the model is stated mathematically as

$$\begin{aligned} \text{Min} \quad & \{E(NORS/X),\ E(SDO/X),\ CX\} \\ \text{s.t.} \quad & X \geq 0, \text{ integer} \end{aligned}$$

This model is also operationally meaningful, since all of these measures highly relate to operations. E(NORS/X) measures the expected number of cannibalized airplanes. E(SDO/X) is an indicator of the additional work load for repairs, and CX gives the cost of the kit.

The following solution strategy is proposed for solving this multiobjective programming problem. First of all, the problem can be considered as a parametric programming in terms of budget levels 'b' as shown below.

$$\begin{aligned} &\text{Min } (E(NORS/X),\ E(SDO/X)) \\ &\text{s.t. } CX \leq b \\ &\qquad X \geq 0, \text{ integer.} \end{aligned}$$

This reduced problem may be further investigated as two single-objective problems as follows:

$$(1) \quad \text{Min} \quad E(NORS/X)$$
$$\text{s.t.} \quad CX \leq b$$
$$X \geq 0, \text{ integer}$$

and

$$(2) \quad \text{Min} \quad E(SDO/X)$$
$$\text{s.t.} \quad CX \leq b$$
$$X \geq 0, \text{ integer}$$

Obviously the optimal solution of (1) is not necessarily optimal to (2), and conversely. If $\bar{X}^*$ solves (1) then $E(SDO/\bar{X}^*)$ turns out to be very high, and conversely if $\tilde{X}^*$ solves (2), then $E(NORS/\tilde{X}^*)$ is very high. This phenomenon has also been observed by WPAF Logistic Command and was stated explicitly in [8], the Preliminary Evaluation of DO29 WRSK Model. Since $E(SDO/X) \geq E(NORS/X)$ for a given kit X, a best compromise solution between the E(NORS/X) and E(SDO/X) is sought for a specified budget level. Figure 4.1 illustrates the relation between E(SDO/X) and E(NORS/X) for specified budget levels.

Problems (1) and (2) can be merged into one problem (3) by the use of a weight w as follows.

$$(3) \quad \text{Min} \quad \{E(NORS/X) + w\, E(SDO/X)\}$$
$$CX \leq b$$
$$X \geq 0, \text{ integer}$$

The weight w expresses the quantity of NORS planes equivalent to each SDO. Prior assessment of the weight is required to obtain the best compromise

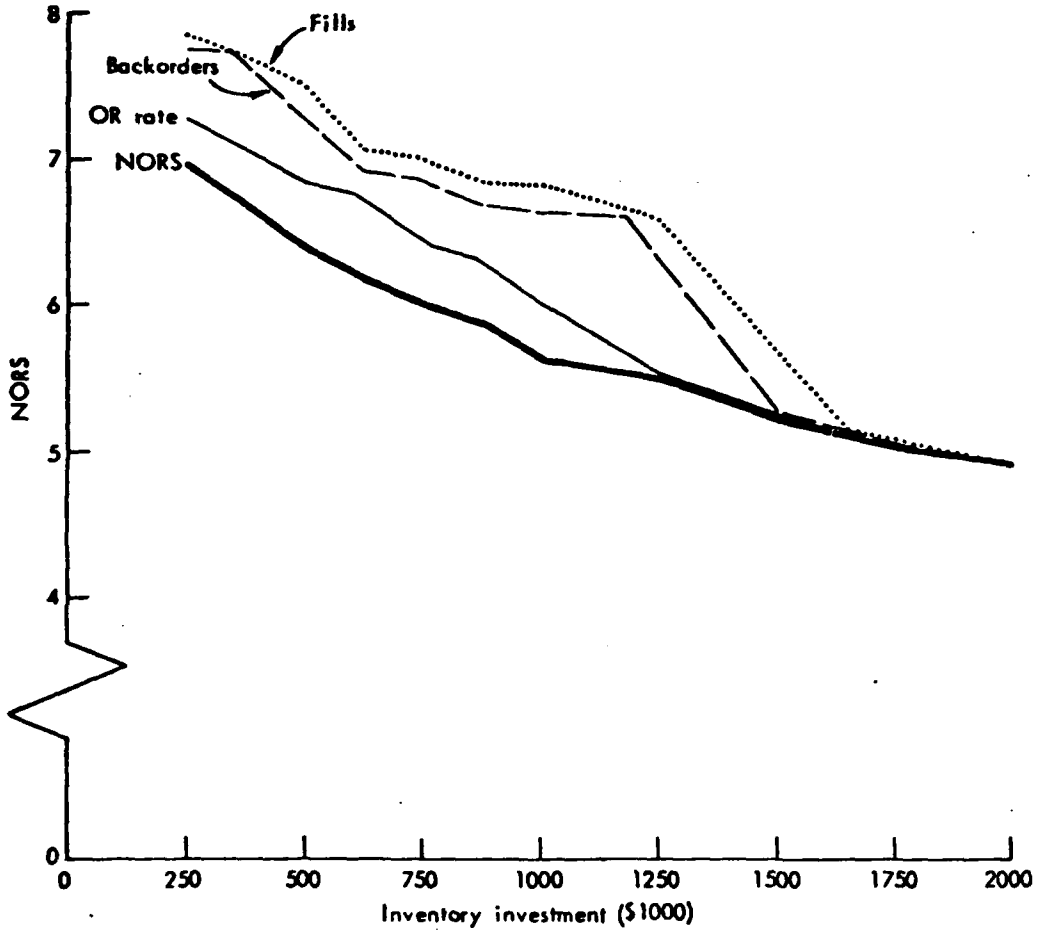


Figure 4.1. Performance Measures Vs. Various Investment Levels (Figure taken from [3])

solution (see Figure 4.2). In the current DO29 Model, the reciprocal of the weight is changed dynamically in order that both objectives converge simultaneously. It is also mentioned in [8] that $1/w = 50$ produces good results. During this research it was observed that $w = 0.025$ produced a good solution for the initial six-day kit and w between 0.015 and 0.025 produced good results for the 30-day kit. A detailed description of selecting a value for w will be provided later when we discuss in Part V the computational experience with this algorithm.

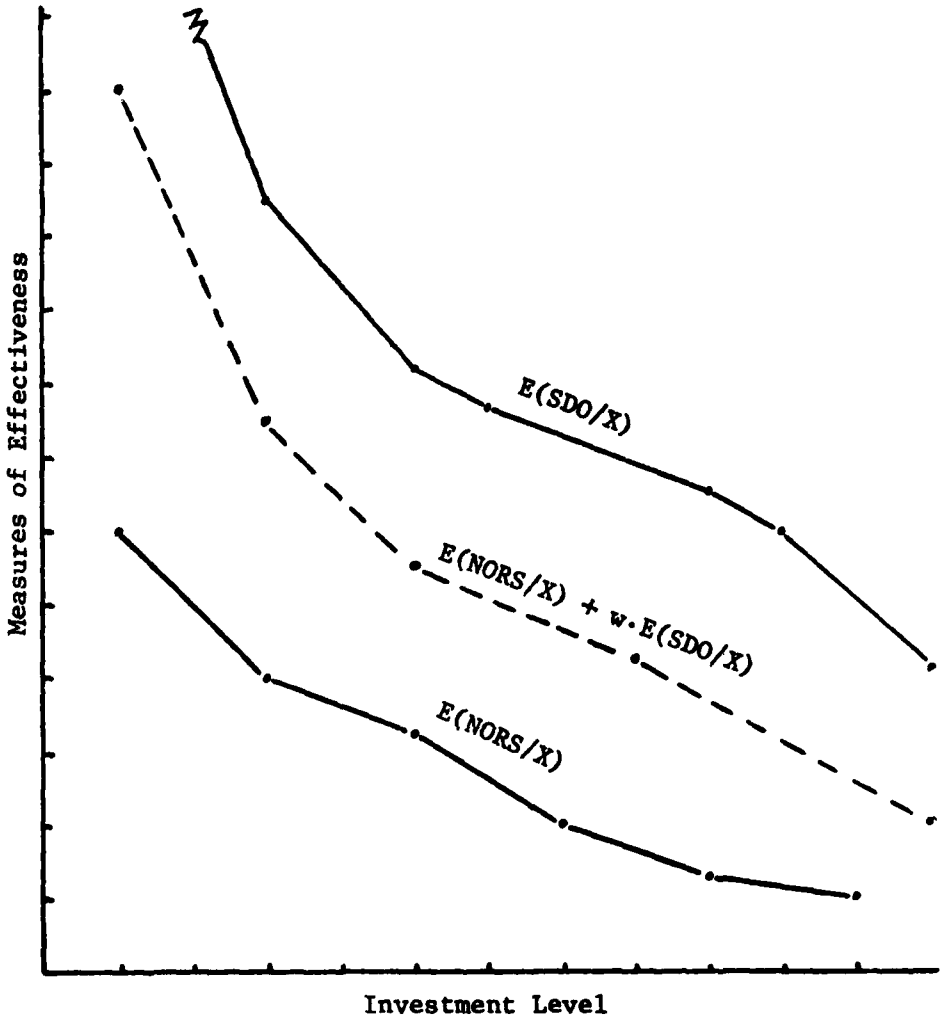


Figure 4.2. Compromise Solution For a Given Weighting Factor w

Potential Solution Methods

Many potential solution methods were investigated. Those that were found significant to the solutions of problem (1), (2), and (3) are summarized as follows:

A Primal Search Algorithm

The problem to be solved is of the form

$$\begin{aligned} \text{Min} \quad & f(X) = E(NORS/X) \\ \text{s.t.} \quad & CX \leq b \\ & X \geq 0 \end{aligned}$$

The discrete convexity of the (NORS/X) function was proved by B. L. Miller [6]. The Lagrangian for this problem is given by

$$L(X;\lambda) = f(X) + \lambda(CX - b)$$

and

$$\Delta_i L(X;\lambda) = L(X + e_i;\lambda) - L(X;\lambda) \qquad i = 1,2,\ldots,n$$

where e_i is the ith unit rector.

This algorithm makes use of the discrete convexity (see [6] and [9]) and the montonicity. A better feasible solution is obtained from the current solution by determining a search direction and heuristically selecting a step size.

The procedure is as follows:

Step 0. Initialization, pick a λ and p where $0 < p < 1$, and set $k = 0$. Find X^o such that $L(X^o;\lambda) \leq L(X^o \pm e_i;\lambda)$ for $i = 1, \ldots, n$. a_i's are assumed to be 1 for simplicity. The following reasoning can easily be extended to the cases with $a_i > 1$. In particular, the first differences for the function are given by

$$\Delta_i L(X; \lambda) = L(X + e_i; \lambda) - L(X; \lambda) \text{ for } i = 1, \ldots, n \qquad (3.7)$$

and

$$\Delta_i L(X; \lambda) = -\sum_{j=0}^{m} ((F_i(x_i + j + 1) - F_i(x_i + j)) \prod_{\substack{k=1 \\ k \neq i}}^{n} F_k(x_k + j) - 1 +$$

$$F_i(x_i + m + 1) + \lambda c_i$$

Since $0 \leq \prod_{k \neq i} F_i(x_i + j) \leq 1$ for $0 \leq j \leq m$, then

$$-\sum_{j=1}^{m} F_i(x_i + j) - F_i(x_i + m + 1) + F_i(x_i) + \sum_{j=1}^{m} F_i(x_i + j) - 1 +$$

$$F_i(x_i + m + 1) + \lambda c_i \leq \Delta_i L(X; \lambda)$$

and

$$F_i(x_i) - 1 + \lambda c_i \leq \Delta_i L(X; \lambda) \leq F_i(x_i + m + 1) - 1 + \lambda c_i$$

Therefore, it suffices to find an $X^o = (x_1^o, \ldots, x_n^o)$ where each x_i^o satisfies $-1 + F_i(x_i^o - 1) + \lambda C_i \leq 0$ and

$$-1 + F_i(x_i^o) + \lambda c_i > 0 \quad i = 1, \ldots, n.$$

Step 1. Search for a better feasible solution; set $k = k + 1$.

Construct a favorable direction

$$d_k = -\nabla f(X^k) - p \cdot p_k \cdot C$$

where $\nabla f(X^k) = (\Delta_1 f(X^k), \ldots, \Delta_n f(X^k))$, $\Delta_i f(X^k) =$

$$-\sum_{j=0}^{m} \left[F_i(x_i^k + j + 1) - F_i(x_i^k + j)\right] \prod_{\substack{\ell=1 \\ \ell \neq i}}^{n} F_i(x_\ell^k + j) - 1 + F_i(x_i^k + m + 1)$$

and p_k is obtained by solving $(-\nabla f(X^k) - p_k C).C = 0$ and p is left as an input parameter by the user. d_k is normalized such the $\max\{|d_{ki}|; i = 1, \ldots, n\} = 1$. Calculate an α by $\alpha = \left[\frac{M - C.X^k}{\sqrt{n}\ C.d_k}\right]$ where $[\cdot\cdot]$ denotes the greatest integer function. Then a point y_α is obtained as $y_\alpha = X^k + \alpha d_k$. The nearest integer point V to y_α is obtained by $v_i = [y_{\alpha i} + .5]$ for $i = 1, \ldots, n$.

i) If V is feasible and $f(V) < f(X^k)$, set $X^{k+1} = V$, to to Step 1.

ii) If V is feasible, but $f(V) \geq f(X^k)$, then check the points $V + e_i$ for $i = 1, \ldots, n$, Figure 4.3. If there exists an i* such that $V + e_{i*}$ is feasible and $f(V + e_{i*}) < f(X^k)$, set $X^{k+1} = V + e_{i*}$, and go to Step 1. Otherwise, go to Step 2.

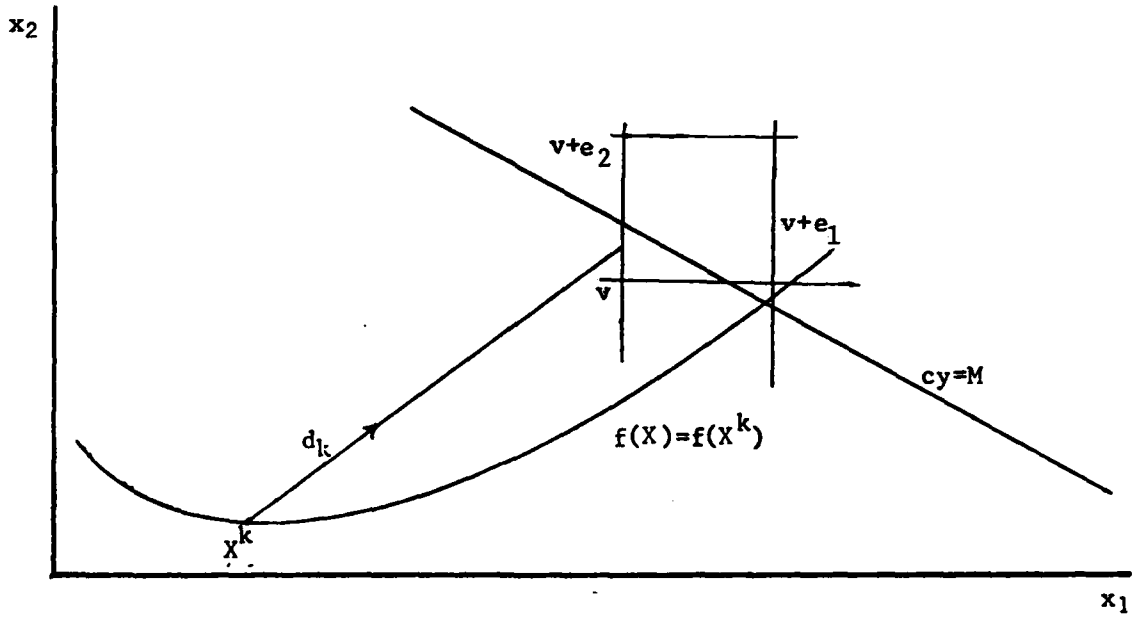


Figure 4.3. Search in the Positive Quadrant

iii) If V is not feasible, then search the points $V - e_i$ for $i = 1, \ldots, n$. If there exists an i^* such that $V - e_{i^*}$ is feasible and $f(V - e_{i^*}) < f(X^i)$, set $X^{k+1} = V - e_i$ and go to Step 1. Otherwise go to Step 2.

Step 2. Test of optimality. Test the points $X^k + e_i$, $i = 1, \ldots, n$. If there exists an i^* such that $X^k + e_{i^*}$ is feasible and $f(X^k + e_{i^*}) < f(X^k)$, set $X^{k+1} = X^k + e_{i^*}$ and go to Step 1, otherwise terminate.

The Modified Algorithm of Brooks et al.

This approach requires the objective function f(X) be approximated by a separable function of the form

$$f(X) \simeq \sum_{i=1}^{n} \sum_{j=0}^{m} b_j F_i(x_i; j)$$

where

$$b_j = \beta_j(x_1', x_2', \ldots, x_n') \geq 0$$

for some $X' = (x_1', x_2', \ldots, x_n')$, and for some function β_j (.).

Let $$g_i(x_i) = \sum_{j=0}^{m} b_j F_i(x_i; j)$$

then

$$f(X) = \sum_{i=1}^{n} g_i(x_i)$$

When $g_i(X_i)$ is strictly decreasing, then the implementation of marginal analysis is justified, and leads to the following series of steps.

Step 0. Initialization, pick $b_0^0, b_1^0, \ldots, b_m^0$, set k = 1.

Step 1. Solve the problem.

$$\min \sum_{i=1}^{n} g_i(x_i)$$

Subject to

$$\sum_{i=1}^{m} c_i x_i \leq b$$

$$x_i \geq 0 \text{ integer}$$

by the following algorithm which is known as marginal analysis [4].

Step 1 i) Initialization: $X^0 = (0, 0, \ldots, 0)$, set $\ell = 0$.

Step 1 ii) $Y^{\ell+1} = Y^{\ell} + e_j$, if feasible, where j is any index for which the ratio

$$r_i = \frac{g_i(y_i + 1) - g_i(y_i)}{c_i}$$

is maximum.

Step 1 iii) Stop if $CY^{\ell+1} \geq b$, and set $X^k = Y^{\ell}$, then go to Step 2. Otherwise go to Step 1 ii.

Step 2 Calculate $b_j^k = \beta_j(X^k)$ for j = 0, 1, ..., m.

Step 3 If b_j^k is very close to b_j^{k-1} for all j = 0, 1, ..., m, then $X^* = X^k$, and stop. Otherwise, set k = k + 1, and go to 1.

This is a modified version of the algorithm proposed by R. B. S. Brooks, C. A. Gillen, and J. Y. Lu on page 24 of [3]. The Step 1 of their

approach was quite different from the one described above. They attempted to find the saddle point solution to the problem

$$\max_{\lambda \geq 0} \min_{X \geq 0} L(X:\lambda) = \Sigma g_i(x_i) + \lambda(\Sigma c_i x_i - b).$$

Their Step 1 was as follows:

Step 1) Find $x_1^k, x_2^k, \ldots, x_n^k$ at k^{th} iteration which maximizes

$$\sum_{j=0}^{m} b_j \sum_{i=1}^{n} \log F_i(x_i + j) - \lambda \sum_{i=1}^{n} c_i x_i$$

Since this function is separable, the optimization is carried out for each component separately. At the end of the third step, the Lagrange Multiplier is rearranged as suggested in [2], and the algorithm repeated again.

A Saddle Point Search Algorithm

This algorithm solves the problem

$$\max_{\lambda > 0} \min_{X \in \Pi^n} L(X;\lambda).$$

The algorithm in fact searches the saddle point (X^*, λ^*) of the Lagrangian function. This is, however, more complicated than the previous algorithms, and requires subalgorithms to solve the primal and dual subproblems. This was originally suggested by H. Everett [2].

The procedure is as follows:

Step 0. Initialization. Pick X^o and λ_o. Set k = 0.

Step 1. Solve the primal subproblem.

$$\min_{X \in \Pi^n} L(X;\lambda_k)$$

to obtain X^{k+1}.

Step 2. Solve the dual subproblem

$$\max_{\lambda>0} L(X^{k+1};\lambda)$$

to obtain λ_{k+1}.

Step 3. Stop if λ_{k+1} is very close to λ_k. Otherwise set k = k + 1 and go to Step 1.

B. L. Miller developed an algorithm to solve the primal subproblem for discretely convex functions. Since a detailed description of this algorithm is presented in [6], it is not going to be discussed here. This is a very complicated algorithm. The author claims it produces good solutions.

At the k^{th} iteration ($k \geq 1$), a piecewise convex approximation of $L(X;\lambda)$ can be given in the space generated by the points, X^0, X^1, ..., X^{k+1} as follows:

$$L(X;\lambda) = \sum_{t=0}^{k+1} \mu_t L(X^t;\lambda)$$

where $\sum_{t=0}^{k+1} \mu_t = 1$, $\mu_t \geq 0$ for all t = 0, 1, ..., k+1.

Then the dual subproblem is restated as

$$\max \sum_{t=0}^{k+1} \mu_t L(X^t;\lambda)$$

Subject to

$$\sum_{t=0}^{k+1} \mu_t = 1$$

$$\lambda>0,\ \mu_t \geq 0 \text{ for } t = 0, 1,\ldots, k+1$$

Since $L(X;\lambda) = f(X) + \lambda\, CX - \lambda b$, then the problem

$$\max \sum_{t=0}^{k+1} \mu_t f(X^t) + \lambda \sum_{t=0}^{k+1} \mu_t CX^t - \lambda b$$

$$\text{Subject to} \quad \sum_{t=0}^{k+1} \mu_t = 1$$

$$\lambda > 0,\ \mu_t \geq 0 \text{ for } t = 0, 1, \ldots, k+1$$

can be considered as the Lagrangian of the linear program

$$\min \sum_{t=0}^{k+1} \mu_t f(X^t)$$

$$\text{Subject to} \quad \sum_{t=0}^{k+1} \mu_t CX^t \leq b$$

$$\sum_{t=0}^{k+1} \mu_t = 1$$

$$\mu_t \geq 0 \text{ for } t = 0, 1, \ldots, k+1$$

or as the Lagrangian of the dual linear program

$$\min\ (\lambda b + \nu)$$

$$\text{Subject to} \quad \lambda CX^t + \nu \geq f(X^t) \text{ for } t = 0, 1, \ldots, k+1$$

$$\lambda > 0$$

$$\nu \text{ unrestricted}$$

Therefore the dual linear program can be solved instead of the subproblem in Step 2. This can be done as suggested by Brooks R. B. S. and A. M. Geoffrion in [2].

Example Problem Solutions

General

A war readiness spares kit problem with five items is used to demonstrate the calculations in the algorithms. Computer program listings are given in Appendix A. The steady-state Poisson demand rates and cost of each item is given in Table 4.1.

The budget is assumed to be M = \$25000. In addition the value for m of Equation is chosen to be 5. Also, $a_i = 1$ for i = 1, 2, 3, 4, 5.

TABLE 4.1

COSTS AND DEMAND RATES FOR THE SAMPLE WRSK PROBLEM

Item No.	Cost/Unit	Demand Rate
1	2980.	2.10
2	1751.	1.50
3	462.	1.20
4	1500.	5.00
5	345.	3.50

Solution by Means of the Primal Search Method

Iteration 0. Initialization: Parameter p is chosen to be .85 and the Lagrange multiplier is specified as $\lambda = .0002$. (For further reference see [9].) This iteration seeks an X^o which satisfies the condition of 3.11, page 4.6.

Since $-1 + F_1(1) + (.0002)\ (2980) = -1 + .37962 + .596 = -.02438$, and $-1 + F_1(2) + (.0002)\ (2980) = -1 + .64963 + .596 = .24563$, $x_1^o = 2$. Repeating the same argument for each component, yields $X^o = (2, 2, 3, 6, 6)$.

Iteration 1. Seeking a better feasible solution: The first differences, $\nabla f(X^k)$, at the point X^o = (2, 2, 3, 6, 6) are computed (by Step 1, page 4.6), and Δf (2, 2, 3, 6, 6) = (-.240, -.106, -.015, -.163, -.035)

$$P_1 = \frac{-\Delta f(2, 2, 3, 6, 6)}{C.C} .C = .00008$$

Since p = .85, then after normalization, the direction is

$$d_1 = (.610, -.225, -.268, 1., .190)$$

and α is evaluated to be 1. Thus a point U = (3, 2, 3, 7, 6) is obtained by finding the nearest integer point to Y = (2.610, 1.775, 2.732, 7., 6.190). However the point U is not feasible, cost (U) = 26398, hence a search in the negative direction is required. For that purpose $\Delta_1 f(2, 2, 3, 7, 6) = -.262$, $\Delta_2 f(3, 1, 3, 7, 6) = -.333$, $\Delta_3 f(3, 2, 2, 7, 6) = -.076$, $\Delta_4 f(3, 2, 3, 6, 6) = -.185$, and $\Delta_5 f(3, 2, 3, 7, 5) = -.100$ are calculated. The minimum of these values is for i = 3, but (3, 2, 2, 7, 6) is not feasible. The second minimum value is for i = 5, again (3, 2, 3, 7, 5) is not feasible. The third minimum is for i = 4, fortunately (3, 2, 3, 6, 6) is feasible and f(3, 2, 3, 6, 6) = .98571. Therefore X^1 = (3, 2, 3, 6, 6) with CX = 24898.

Iteration 2. As in Iteration 1, a search direction d_2 = (-.356, .362, -.037, 1., .216) issuing from (3, 2, 3, 6, 6) is determined. A point U = (3, 2, 3, 7, 6) is obtained similarly. Since U is infeasible, and a search in the negative direction does not yield a distinct and better feasible point than X^1 = (3, 2, 3, 6, 6), a search in the positive direction at X^1 is employed. All the points of the form $X^1 + e_i$ for i = 1, 2,..., 5 are infeasible. Hence X^1 = (3, 2, 3, 6, 6) is optimal. This small problem takes .24 seconds of execution time.

Solution by the Modified Solution Method of Brooks et al.

Iteration 1. Choose $b_0 = b_1 = \ldots = b_4 = 0$ and $b_5 = 1$. Set $x_i^o = 0$ for $i = 1, 2, \ldots, 5$, then $g_i(x_i) = b_5 \log F_i(x_i + 5)$, and calculating the ratio $r_i = \dfrac{g_i(x_i + 1) - g_i(x_i)}{C_i}$ for each $i = 1, \ldots, 5$ yields: $r_1 = 0.00000496$, $r_2 = 0.00000202$, $r_3 = 0.00000270$, $r_4 = 0.00014200$, $r_5 = 0.00024952$. For instance,

$$r_1 = \frac{b_5}{c_1} \log \frac{F_1(6)}{F_1(5)} = \frac{1}{2980} \log \frac{.99413}{.97955} = 4.96 \times 10^{-6}$$

r_5 is the maximum, hence $x_5^1 = 1$. Updating r_5 yields: $r_1 = 0.00000496$, $r_2 = 0.00000202$, $r_3 = 0.00000270$, $r_4 = 0.00014200$, $r_5 = 0.00011714$. The maximum is r_4, then $x_4^1 = 1$. Continuing in this fashion produces $X^1 = (2, 2, 2, 8, 7)$ with $\text{cost}(X^1) = 24801$, and calculating b_j by $b_j^1 = \prod_{i=1}^{n} F_i(x_i^1 + j)$ gives $b_0^1 = .419$, $b_1^1 = .726$, $b_2^1 = .898$, $b_3^1 = .967$, $b_4^1 = .991$, and $b_5^1 = .998$.

Iteration 2. Repeating the same steps with b_j^1 $j = 0, 1, \ldots, 5$ yields $X^2 = (2, 2, 4, 7, 9)$ with $\text{cost}(X^2) = 24915$, and $b_0^2 = .450$, $b_1^2 = .728$, $b_2^2 = .891$, $b_3^2 = .962$, $b_4^2 = .988$, and $b_5^2 = .996$. Since b_j^1's are not within .001 of b_j^2's, go to Iteration 3.

Iteration 3. Repeating the same steps again with b_j^2 for $j = 0, 1, 2, \ldots, 5$ yields $X^3 = X^2$ and $b_j^3 = b_j^2$, hence X^2 is optimal with $f(2, 2, 4, 7, 9) = .98619$.

This solution however is a near optimal solution. The primal search procedure has produced a better solution than this one. On the other hand this method is faster, the execution time to solve this small problem with this algorithm is .15 seconds.

In this algorithm b_j's are treated as parameters. They have practical implications for real life problems. β_j is given by the product $\prod_{i=1}^{n} F_i(\bar{X} + j)$ for some $\bar{X}$ and $j = 0, 1, \ldots, m$. This product is in fact equal to the probability of cannibalizing at most j airplanes with a kit $\bar{X}$. In other words b_j gives the operational rate given j airplanes for cannibalization for a kit $\bar{X}$. Selecting $b_0 = b_1 = \ldots = b_{m-1} = 0$ and $b_m = 1$ implies that all the planes will be cannibalized and corresponds to the most pessimistic view.

Solution by the Saddle Point Search Algorithm

Iteration 1. A starting solution of $X^0 = (10, 10, 10, 10, 10)$ with $f(X^0) = .02486$ and cost $(X^0) = 70380$ and a Lagrange multiplier of $\lambda_0 = .0001$ is chosen. In the first step the subproblem $\min_{X \in \Pi^n} L(X;\lambda_0) = f(X) + \lambda_0(CX - M)$ is solved by the algorithm proposed by B. L. Miller in [6]. This algorithm produces the solution $X^1 = (2, 2, 2, 6, 6)$ with $f(X^1) = 1.28241$ and $\text{cost}(X^1) = 21456$.

Step 1. Initialization: An initial point is obtained by solving $-1 + F_i(\omega_i - 1) + \lambda C_i \leq 0$ and $-1 + F_i(\omega_i) + \lambda C_i > 0$ and setting $X_i = \max(0, \omega_i - k)$ for all $i=1, \ldots, n$ where k is an input parameter and chosen as 1 in this particular example. Since $-1 + F_1(2) + (.0001)(2980) = -1 + .6496 + .2980 = -.0524$ and $-1 + F_1(3) + (.0001)(2980) = -1 + .8386 + .2980 = .1366$, then $\omega_1 = 3$, consequently $x_1^o = 2$. Repeating the same argument for each component yields that $x_2^o = 2$, $x_3^o = 2$, $x_4^o = 6$, and $x_5^o = 6$.

Step 2. Finding A Stationary Vector: Starting from $X^o = (2, 2, 2, 6, 6)$, a point X is sought such that $\Delta_i g(X - e_i) \leq 0$ and $\Delta_i g(X) > 0$ for all $i=1, \ldots, 5$. For instance for item 5, $\Delta_5 g(2, 2, 2, 6, 6)$ is

calculated as follows; $\Delta_5\, g(2, 2, 2, 6, 6) = \lambda\, C_5 + F_5(6 + 5 + 1) - 1.$

$$- \sum_{j=0}^{5} \left(F_5(6 + j + 1) - F_5(6 + j)\right) \prod_{i=1}^{4} F_i(X_i + j)$$

j	$F_1(2+j)$	$F_2(2+j)$	$F_3(2+j)$	$F_4(6+j)$	$F_5(7+j) - F_5(6+j)$	Product
0	.64963	.80885	.87949	.76218	.03855	.01358
1	.83864	.93436	.96623	.86663	.01687	.01107
2	.93787	.98142	.99225	.93191	.00656	.00558
3	.97955	.99554	.99850	.96817	.00230	.00217
4	.99414	.99907	.99975	.98630	.00073	.00072
5	.99851	.99983	.99996	.99455	.00021	.00020
					SUM =	.03332

Then $\Delta_5\, g(2, 2, 2, 6, 6) = .0345 + .99992 - 1. - .03332 = .00110.$ Since it is positive, $\kappa_5 = 6$. This argument is repeated for each item to obtain a stationary vector X = (2, 2, 2, 6, 6) with f(X) = 1.28241, cost (X) = 21456, and g(X) = 3.42801.

Step 3. A heuristic approach to obtain improvement in "up-phase": Let $S_i = \sum_{k=i}^{n} e_k - e_i$, then the ratios $\Delta_i\, g(X + S_i)/\Delta_i\, g(X)$ are ranked in ascending order, and the vectors of the form $X + U_i$ will be tested for improvement where U_i is the vector with +1 for variables ranked from 1 to i and 0 for the ones ranked i + 1 to n. Calculation of the ratios gives $\Delta_1\, g(X + S_1)/\Delta_1\, g(X) = -5.61$, $\Delta_2\, g(X + S_2)/\Delta_2\, g(X) = .15$, $\Delta_3\, g(X + S_3)/\Delta_3\, g(X) = 1.47$, $\Delta_4\, g(X + S_4)/\Delta_4\, g(X) = 1.44$ and $\Delta_5\, g(X + S_5)/\Delta_5\, g(X) = 1.64$. The ordering according to the magnitudes yields 1, 2, 4, 3, 5 respectively. For instance, for i = 4 U_4 = (1, 1, 0, 1, 0) and $X + U_4$ = (3, 3, 2, 7, 6), but this point does not yield a better solution, since $g(X + U_4) = 3.48107$ is greater than g(X) = 3.42801.

Repeating the same argument for each component results that the current solution is stationary.

The same argument is repeated for the "down-phase" too. This does not yield a better solution either.

Step 4. A Second Initialization and Reducing the Set of Potential Improving Points.

First the up-phase is implemented. For instance for i = 1, the vector (0, 1, 1, 1, 1) will be checked for improvement by looking at Δ_1 g(2, 3, 3, 7, 7). Since Δ_1 g(2, 3, 3, 7, 7) = -.028 is negative, the test fails. For i = 2, the vector (1, 0, 1, 1, 1) is tested and Δ_2 g(3, 2, 3, 7, 7) = .005, then the conclusion is that x_2^* can not be 3. Next the vector (1, 0, 0, 1, 1) is used for the third component and Δ_3 g(3, 2, 2, 7, 7) = -.056. Similarly Δ_4 g(3, 2, 3, 6, 7) = -.069 and Δ_5 g(3, 2, 3, 7, 6) = -.022. The test fails for i = 1, 3, 4, and 5. Consequently Part 1 is used. Calculation of the ratios Δ_1 g(2, 3, 3, 7, 7)/Δ_1 g(2, 2, 2, 6, 6) = -5.61, Δ_3 g(3, 2, 2, 7, 7)/Δ_3 g(2, 2, 2, 6, 6) = 1.37, Δ_4 g(3, 2, 3, 6, 7)/Δ_4 g(2, 2, 2, 6, 6) = 1.33 and finally Δ_5 g(3, 2, 3, 7, 6)/Δ_5 g(2, 2, 2, 6, 6) = 1.52 gives that the maximum ratio is for i = 5. By contradiction it will be proved that s_5 = 1 implies s_3 = 1 where X + S is the point to be tested. Since $b_0 = -\lambda C_3 - \Delta_3$ g(2, 2, 3, 6, 7) = .042 > 0, S_3 can not be 1. Repeating this argument for i = 1 and i = 4 does not yield a better solution. Consequently the down phase is implemented. Down-phase does not generate a better feasible solution. Finally the conclusion is that X = (2, 2, 2, 6, 6) is a stationary point.

Then the linear program

$$\min .02486\,\mu_1 + 1.28241\,\mu_2$$

Subject to $$70380\,\mu_1 + 21456\,\mu_1 \leq 25000$$

$$\mu_1 + \mu_2 = 1$$

$$\mu_1, \mu_2 \geq 0$$

is solved to obtain $\lambda_1 = .0000257$. Since λ_1 is not within .000005 of λ_0 another iteration is performed.

Iteration 2. The subproblem $\min_{X \in \Pi^n} L(X;\lambda_1) = f(X) + \lambda_1(CX - M)$ is solved again in the first step by the same method to obtain $X^2 = (4, 4, 4, 9, 9)$ with $f(X^2) = .17727$ and $\text{cost}(X^2) = 37377$. In the second step the linear program $\min .02486\,\mu_1 + 1.28261\,\mu_2 + .17727\,\mu_3$,

Subject to $$70380\,\mu_1 + 21456\,\mu_2 + 37377\,\mu_3 \leq 25000$$

$$\mu_1 + \mu_2 + \mu_3 = 1$$

$$\mu_i \geq 0 \quad i = 1, 2, 3$$

is solved to obtain $\lambda_2 = .0000694$. Since λ_2 is not within .000005 of λ_1, another iteration is required. Continuing in this fashion produces the following results:

Iteration 3. $X^3 = (3, 3, 3, 8, 7)$ with $f(X^3) = .49955$ and cost $(X^3) = 29994$. Further $\lambda_3 = .0000917$.

Iteration 4. $X^4 = (2, 2, 3, 7, 7)$ with $f(X^4) = 1.02459$ and cost $(X^4) = 23763$. Also $\lambda_4 = .0000848$.

Iteration 5. $X^5 = (3, 2, 3, 7, 7)$ with $f(X^5) = .75556$ and cost $(X^5) = 26743$. Also $\lambda_5 = .0000903$.

Iteration 6. $X^6 = (2, 2, 3, 7, 6)$ with $f(X^6) = 1.06282$ and cost $(X^6) = 23418$ and $\lambda_6 = .0000903$.

This algorithm produced a solution with a better objective value than the previous ones, but it is an infeasible solution. In fact, this algorithm suffers from the duality gap. In the first step of this algorithm, the subproblem $\min_{X \in \Pi^n} L(X;\lambda)$ is solved. Since $L(X;\lambda)$ is discrete in $X \in \Pi^n$, there is a range for λ values which produce the same answer. From the results of previous algorithms, it was discovered that the optimal solution was $X^* = (3, 2, 3, 6, 6)$. Evidently its corresponding Lagrange multiplier should lie between $\lambda_4 = .0000848$ and $\lambda_5 = .0000903$. Unfortunately the algorithm won't generate this value, and if λ is close to λ_4, the solution to subproblem is X^5 and if λ is close to λ_5, then the solution is X^6, see Figure 4.4.

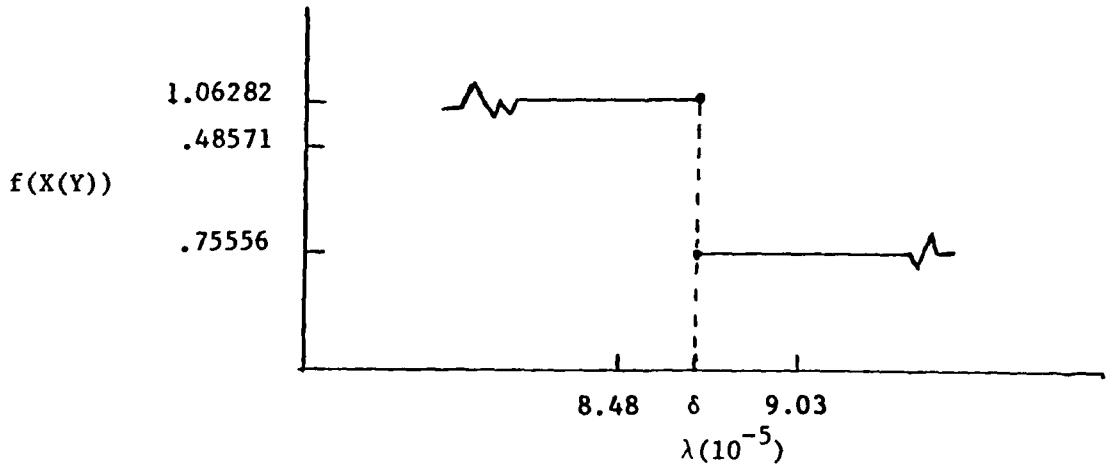


Figure 4.4. Duality Gap

Selection of Solution Method

The objective of this research is to develop and/or modify an efficient solution method to solve the WRSK problem. Therefore, these three solution methods have been under investigation to observe their computational requirements. The result of this investigation shows that the Modified Algorithm of Brooks et al. is more efficient in terms of computational time than the others. This algorithm is also going to be compared with the current DO29 Model which is summarized on the next page. The result of this comparison will be discussed in Section V.

The Modified Algorithm of Brooks et al. is a greedy algorithm in nature, and will be refined to in the remainder of this report as the "Greedy Algorithm". It uses marginal analysis as a subalgorithm. It is easier to implement in solving the WRSK problem and is faster than the others. The selection of its initial parameters does not require any guesswork from the user. The parameter values obtained for the requirement kit 57-18 can serve that purpose. On the other hand, marginal analysis adds one unit of the item which will yield the highest rate of return in terms of performance measures per dollar invested in the kit. Approximation of the E(NORS/X) function by a separable function [as discussed in Part III] eases the computations and in fact eliminates the need for updating all of the marginal returns whenever a unit of some item is added to the kit as is necessary in the current DO29 computer program. The speed of convergence of an algorithm is very important, especially when solving large size problems, and this algorithm supplies this advantage.

The others have some particular advantages, but they are slower than this algorithm. The Saddle Point Search Algorithm is theoretically very

appealing but suffers from the duality gap as indicated by the small example of the last section. The Primal Search Algorithm uses the first differences in determining the search direction to explore better feasible solutions. All three algorithms contain some heuristic rules. Thus the solutions are not necessarily the true optimum which, for the same reason, is true for the current DO29 Model. The comparison of their solutions either with each other or with the DO29 solution indicates that they can generate good near optimal solutions.

The Current DO29 WRSK Model

This model and its associated computer program is currently in use by WPAFLC. This solution method also makes use of marginal analysis. A detailed description of this algorithm is provided in [8]. A brief description of the algorithm is provided here. The algorithm solves the problems of the form

$$\begin{aligned} & \text{Min } CX \\ \text{s.t.} \quad & E(NORS/X) \leq \alpha_1 \\ & E(SDO/X) \leq \alpha_2 \\ & X \geq 0 \quad \text{integer} \end{aligned}$$

in the following manner.

Step 0. Set $U = 0$, $X^o = 0$

Step 1. Choose a W^k and I^k_{Nt}

Step 2. Add I_{Nt}^{k} units to the kit by selecting the highest I_{Nt}^{k} ratios from

$$r_i = \frac{W^k \Delta_i \, E(NORS/X^k) + \Delta_i \, E(SDO/X^u)}{C_i}$$

$$X^{k+1} = X^k + \sum_{ie H_{I_{Nt}}} ei$$

where $H_{I_{Nt}}^{k}$ is the index set of the highest I_{Nt}^{k} ratios.

Step 3. Calculate $E(NORS/X^{k+1})$ $E(SDO/X^{k+1})$ and CX^{k+1}

Step 4. Phase 1: Terminate if $CX^{k+1} > b^*$

where b^* is 90% of the cost of Kit 57-18

Phase 2: Terminate if $E(NORS/X^{k+1}) \leq \alpha_1$ and $E(SDO/X^{k+1}) \leq \alpha_2$

Otherwise Set k = k + 1 and go to Step 1.

PART V

A SUMMARY OF COMPUTATIONAL EXPERIENCE

Introduction

Computational experimentation was devoted to the investigation of computational requirements, initialization of parameters for the Greedy Algorithm (Modified Algorithm of Brooks et al) and the IBM version of DO29 Program. F4D Data was used in the computational analysis. Some of the subroutines remain the same in both programs in order to make meaningful comparisons. (The FORTRAN listings of the programs are in Appendix B.2 & C.2.) The Greedy Algorithm takes 16.53 seconds of CPU time, and the IBM version of DO29 Program takes 24.78 seconds of CPU time on IBM 370/168 at O.S.U. The Greedy Algorithm is obviously faster on the F4D problem than the algorithm currently in use.

The Parameters b_j and m of the E(NORS/X) Approximation

The Greedy Algorithm requires that b_j and m be initialized in advance. A thorough inspection of the WRSK problem suggested that initial b^o_j values can be calculated from the requirement kit 57-18 by the formula

$$b^o_j = \prod_{i=1}^{n} F_i(\bar{x}_i + ja_i) \qquad j = 0,1,2,\ldots \text{ K = Number of planes and}$$
$$\text{n = Total number of items in consideration}$$

in the squadron, where $\bar{x}_i$ is the kit 57-18 quantity for item i. The initial b^o_j values for Phase 1 and Phase 2 of the WRSK Problem are displayed in Table 5.1. Each b_j is the operational rate given that there are j aircraft for cannibalization. Hence, b^o_j's represent operational rates under the kit 57-18. These are good choices for b_j's since the b^o_j's may be con-

sidered as operational rate goals as a consequence of the AFLCR. Since the WRSK problem attempts to find a kit which will perform better than the kit 57-18, these $b^{o}j$'s, which can be observed in Figures 5.7a and 5.7b on pages 23 and 24 respectively, are the target operational rates for the optimal kit. On the other hand, a natural selection for m is the number of planes in the squadron. An inspection of the Table 5.1 indicates that the b_j's approach 1 for some lower value than m. Consequently, using Table 5.1, one can choose m = 12 for Phase 1 and still get the same answers.

The b^{*}_j's displayed in Table 5.1 correspond to the calculated values from the optimal kit and can also be observed in Figures 5.9a and 5.9b respectively. For Phase 1 the b*j's are very close to the $b^{o}j$'s. However, a similarly close relationship between b*j and $b^{o}j$ does not occur in Phase 2. Note that each $b^{*}j \geq b^{o}j$ for all j. Since the $b^{o}j$ represent the operational rate goals, this shows that the optimal kit provides higher operational rates than the goals.

Instead of hueristically choosing the initial $b^{o}j$'s, the Greedy program of Appendix B.2 uses the $b^{o}j$'s of the 5718 kit. It also uses m=24, the number of planes in the squadron, since 24 is not a large number.

TABLE 5.1

COMPARISON OF bj VALUES

	Phase 1		Phase 2	
j	5718 Kit $b^{o}j$	Optimal Kit b*j	5718 Kit $b^{o}j$	Optimal Kit b*j
0	.00000010	0.0	0.	0.0
1	.00286799	.00172540	0.	0.0
2	.06779460	.06441003	.00000001	.00000935
3	.25850234	.26115723	.00011449	.00376217
4	.48975927	.49893585	.00861967	.05835153
5	.68018747	.69276720	.07484124	.22116812
6	.81107932	.82435903	.23535679	.43810253
7	.89285344	.90472555	.44260779	.63279580
8	.94108090	.95038868	.63162230	.77581119
9	.96842135	.97499286	.77280201	.86953349
10	.98345607	.98772442	.86636330	.92667958
11	.99151443	.99410652	.92392780	.95987318
12	.99573534	.99722461	.95769561	.97850019
13	.99789912	.99871559	.97688397	.98868473
14	.99898552	.99941528	.98754667	.99413854
15	.99951989	.99973806	.99337298	.99700800
16	.99977739	.99988454	.99651327	.99849417
17	.99989891	.99994995	.99818555	.99925267
18	.99995505	.99997868	.99906608	.99963432
19	.99998044	.99999108	.99952462	.99982364
20	.99999167	.99999634	.99976077	.99991621
21	.99999653	.99999852	.99988101	.99996080
22	.99999859	.99999942	.99994153	.99998195
23	.99999944	.99999972	.99997163	.99999182
24	1.	.99999991	1.	.99999636

Articulation of the Weight w

The weighting factor w represents the relative importance of E(SDO/X) function compared to the E(NORS/X) function. The weighting factor is, in fact, a measure of cannibalization of any plane for each back-ordered part. Conversely, $1/w$ measures the number of consolidated parts from a cannibilized airplane. During a mission, if a plane is grounded for lack of spare parts, it may then be cannibalized to repair others by removing its good parts and installing these parts in other downed planes. Theoretically 100% cannibalized is possible. This process increases the work load during the mission. Obviously, a reduction in the work load is required by the mission command. In practice, however, there is empirical knowledge as to when to cannibalize a plane, which plane to cannibalize, and how many parts to be removed from a plane. In the D029 program, manipulation of the parameter RATIO indicates that this empirical knowledge is regularly applied (RATIO in the Greedy program has a different meaning). It has also been observed that RATIO = 50 produces better results than any other value, for the purpose of reaching the E(NORS) and E(SDO) goals simultaneously. The operational meaning of RATIO = 50 is that, on the average, 50 parts are removed from a cannibalized plane; thus RATIO corresponds to $1/w$. This would also mean that 2% of an airplane is cannibalized for each back order. It is suggested by this research that w be assessed in advance by evaluating this empirical knowledge. Tables 5.2, 5.3, 5.4 and Figures 5.1, 5.2, 5.3 show the relationship between E(NORS/X) and E(SDO/X) for different weights. These relationships based on limited experimentation, appear to be linear effects.

The problem is to determine the best effective weight. Computational experimentation with F4D data indicates that w = 2.25% is the effective weight for the initial six-day period, and that $1.5\% \leq w \leq 2.5\%$ gives a range for the effective weight for a thirty-day period. This means that 45 back ordered parts would be cannibalized during the initial six-day and 40 to 65 back ordered parts would be cannibalized during the 30 day mission period. This observation should be combined with the empirical knowledge to come up with an effective weight in determining a best compromise kit.

Only a limited effort was undertaken to determine the underlying relationships between the weighting factor w, budgetary levels, objective function values, and kit configuration sensitivity to these parameters and measures. Tables 5.2, 5.3, 5.4 and Figures 5.1, 5.2, 5.3 indicate various ways that relationships can be exhibited and analyzed. Analysis and study of the relationships exhibited lead to the selection of parameters and measures that are used in the next section which compares various solutions which have been computed.

TABLE 5.2

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w:
PHASE 1 SOLUTIONS FOR A BUDGET OF $3,635,906

$w(10^{-2})$	E(NORS/X)	E(SDO/X)	$\Psi(w)$	$\tilde{\Psi}(w)$
2	4.80602	31.02199	5.42646	5.4346
2.25	4.84981	29.36157	5.51045	5.5061
2.5	4.85617	29.07728	5.58315	5.579
3.	4.87135	28.57339	5.72855	5.724
4.	4.94188	26.69839	6.00982	6.013

$$\Psi(w) = E(NOR/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 28.94220\ w + 4.85578$$

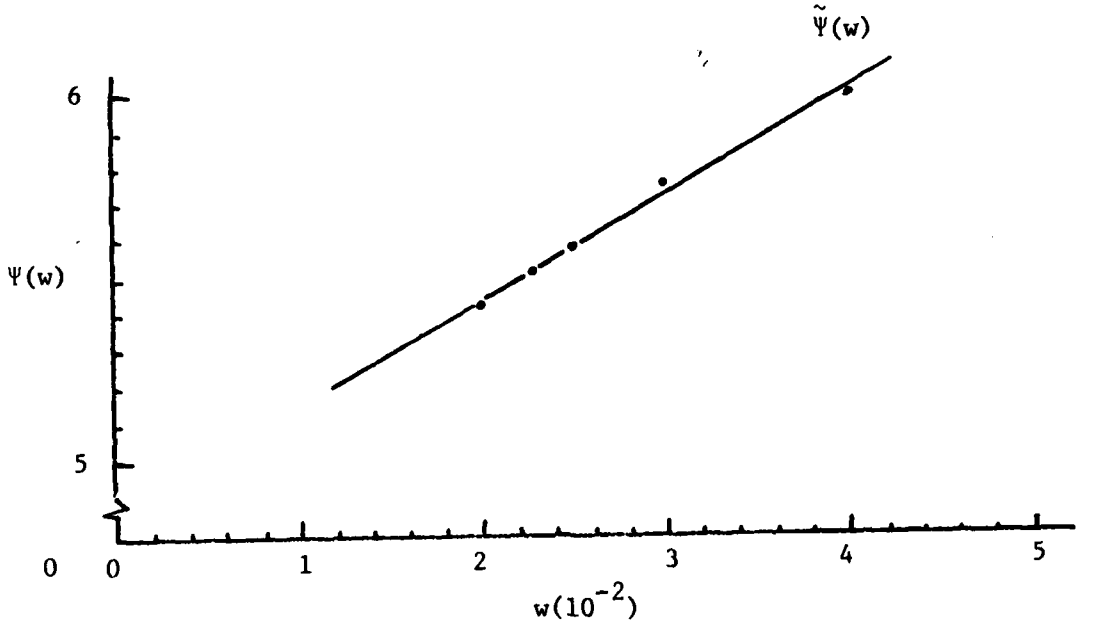


Figure 5.1. Plot of $\tilde{\Psi}(w)$ Exhibited in Table 5.2

TABLE 5.3

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w:
PHASE 2 SOLUTIONS WITH A BUDGET OF $\$7.5 \times 10^6$

$w(10^{-2})$	E(NORS/X)	E(SDO/X)	$\Psi(w)$	$\tilde{\Psi}(w)$
.3	6.56880	261.25884	7.35258	7.36159
1.	6.91434	113.02191	8.04456	8.05923
1.25	7.01317	104.48303	8.31921	8.30838
1.5	7.12255	96.53860	8.57063	8.55754
2.	7.15833	94.41359	9.04660	9.05585
2.5	7.22784	90.83792	9.49879	9.55417

$$\Psi(w) = E(NOR/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 7.06260 + 99.66263w$$

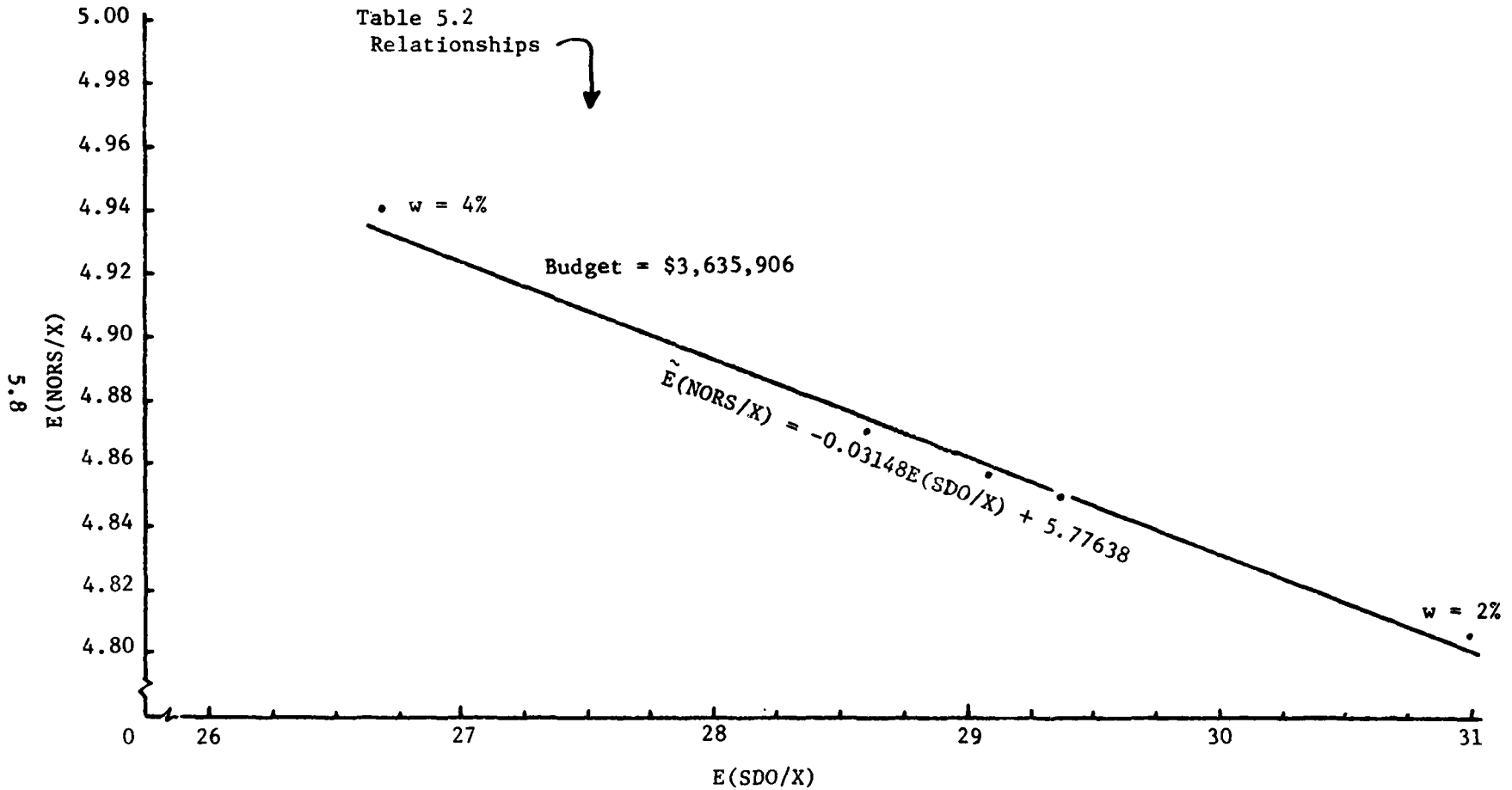
TABLE 5.4

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w:
PHASE 2 SOLUTIONS WITH A BUDGET OF $\$9 \times 10^6$

$w(10^{-2})$	E(NORS/X)	E(SDO/X)	$\Psi(w)$	$\tilde{\Psi}(w)$
1	4.93755	41.56367	5.35319	5.37412
1.5	5.01991	38.17230	5.59249	5.57405
1.75	5.05138	36.31621	5.68691	5.67401
2.	5.08688	34.68301	5.78054	5.77398
2.5	5.12059	33.45378	5.95693	5.97390

$$\Psi(w) = E(NORS/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 4.97427 + 39.98540(w)$$



5.8

Figure 5.2. Plot of E(NORS/X) vs. E(SDO/X) from Table 5.2

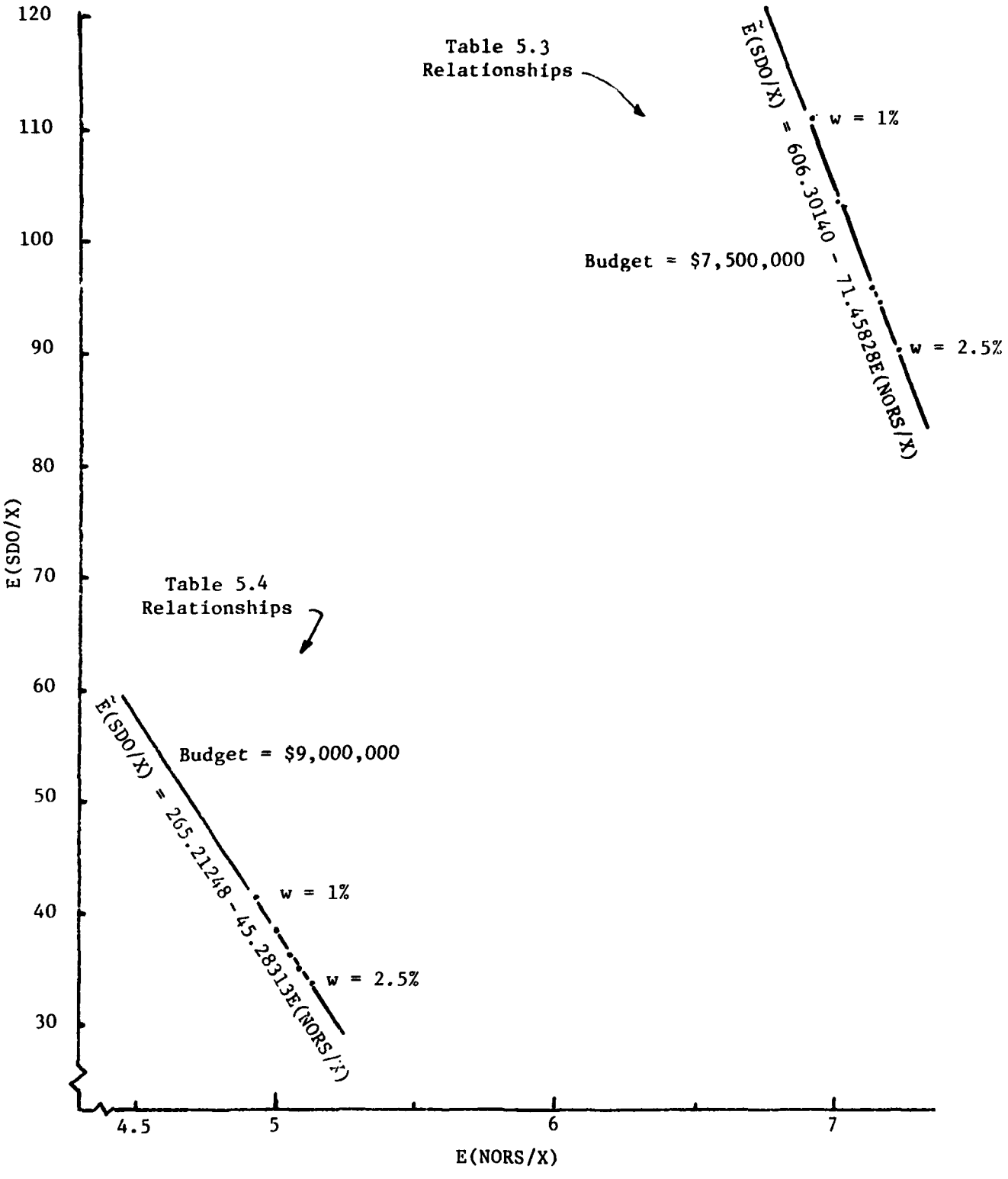


Figure 5.3. Plot of E(SDO/X) vs. E(NORS/X) from Tables 5.3 and 5.4

Comparison of the Solutions

Comparison of solutions was undertaken relative to the two phases of the WRSK Problem. F4D Data was used as sample data, and is provided in Appendix C. Three sets of solutions are shown in Figures 5.4 a, b, 5.5 a, b, and 5.6 a, b. They are the Phase I and Phase II output results of the Greedy Algorithm, the IBM version of the D029 Program and the D029-CDC Program of WPAFLC. The FORTRAN listings of the first two algorithms are provided in Appendix B.2 and C.2, and are the final products of this current research. It should be understood that only limited experimentation has taken place with both of these programs.

The frequency of the differences in item quantities can be of interest in discussing the kits obtained by these algorithms. In Phase I, there are 180 items under consideration. Table 5.5 indicates that 73% of the kit items (for the three comparisons) have the same quantity values with a standard deviation of 1.4%. Also the maximum difference between the individual kit items does not exceed 2. One can also note, for example, that the Greedy Algorithm kit has 40 and 43 less parts in the kit compared to the D029 solutions respectively.

In Phase II, there are 254 items under consideration. Table 5.6 shows the frequency of the discrepancy of the kits. 75% of the individual items in the kits have the same quantity values with a standard deviation of 3%. This indicates that the kits are in close agreement. In this case, the Greedy Algorithm kit has 19 and 18 less parts in the kit compared to the D029 solutions respectively. However, note the difference of nine that occurs in column 1 and column 3.

This difference occurs on item number 249 in Phase II. WPAFLC's D029-CDC Program (Figure 5.6b) yields $x^*_{249}=1$, on the other hand the Greedy Algorithm

TABLE 5.5

FREQUENCY OF DIFFERENCES OF KITS IN PHASE I

Difference	Greedy-DO29 CDC	Greedy-DO29 IBM	DO29 IBM-DO29 CDC
-2	8	-	6
-1	31	46	14
0	134	131	129
+1	7	3	31
Net Δ	-40	-43	+5

TABLE 5.6

FREQUENCY OF DIFFERENCES OF KITS IN PHASE II

Difference	Greedy-DO29 CDC	Greedy-DO29 IBM	DO29 IBM-DO29 CDC
-3	2	—	—
-2	3	3	4
-1	34	37	33
0	196	195	183
1	18	19	30
2	—	3	3
9	1	-	1
Net Δ	-19	-18	+4

(Figure 5.4b) and the IBM Version of the DO29 Program (Figure 5.5b) both produce $x^*_{249}=10$. This item has a moderate demand rate, $\mu_{249}=7.29728$, is not expensive at $653, but has, however, a high number, 15, of applications on an aircraft (Appendix D.2.2). Since $\mu_{249}=7.29728$, probability of demand not exceeding 1 is $P(d \leq 1) = 0.0056$, but probability of demand not exceeding 10 is $P(d \leq 10) = .878$. Addition of 9 more items increases the probability considerably, thus providing more protection from back ordering item 249. The same reasoning applies to other items which have different kit values. If the difference in item quantities is small, the probabilities are close to each other. Investigation has not been undertaken to determine why the CDC version yields $X^*_{249}=1$. Different approaches to calculating the Poisson probabilities may be a contributing factor.

The Greedy Algorithm solution was obtained in 16.53 seconds of CPU time while the DO29-IBM solution was obtained in 24.78 seconds of CPU time. This indicates the potential of a significant improvement in computational time if the Greedy Algorithm is coded for a CDC FORTRAN compiler and used rather than the DO29-CDC program currently in use.

On Figure 5.6b it can be noted that the DO29-CDC program has obtained a kit with E(NORS) = 8.127, E(SDO) = 149.65, and cost = $7,108,451.15. This cost was used as a budget parameter for the Greedy Algorithm solution which is shown on Figure 5.4b. The solution obtained was E(NORS) = 8.037, E(SDO) = 144.96, and cost = $7,108,477. For $4 less expenditure of budget funds, the Greedy Algorithm obtained a kit yielding better performance measures than the DO29-CDC solution.

Figure 5.5b illustrates a DO29-IBM solution for the same budget for $7,562,983 used by the DO29-CDC solution (Figure 5.6b). In this solution

E(NORS) = 8.130, E(SDO) = 145.30 and cost = $7,007,734. The budget is not utilized to its full potential. The objective of the DO29 approach (Part IV, page 4.22) is to meet the NORS and SDO goals in a way that minimizes cost, and under utilization of a budget can result since the actual budget is the last item of priority that is checked in the algorithmic scheme of the DO29 WRSK solution method.

The Greedy solution exhibited in Figure 5.7b for the same budget as in Figures 5.5b and 5.6b ($7,562,983) yields E(NORS) = 7.158, E(SDO) = 96.81, and cost = $7,562,944. All but $39 of the available budget is utilized. These greatly improved performance measures are as expected since more money has gone into the kit.

The WRSK Model solved by the Greedy Algorithm (Part IV, page 4.1) is not the WRSK Model solved by the DO29 Algorithm. For this reason it is difficult at times to make direct comparisons between DO29 solutions and Greedy solutions. It should be recognized again that, in essence, the goals (constraints) to be satisfied by a DO29 solution, are the elements of the objective function (which is to be minimized) of the Greedy model. In like manner, the Greedy model budget constraint, which is not to be exceeded, is the objective function to be minimized in the DO29 model.

Figure 5.8b illustrates an attempt to develop a Greedy solution that is relatively comparable to the DO29-IBM solution of Figure 5.5b. Observe that a budget of $7,075,000 was used leading to a solution in which E(NORS) = 8.173, E(SDO) = 145.09, and cost = $7,074,907 which is within 1% of the DO29-IBM solution of Figure 5.5b. Since two different models are being used, answers exactly equal to each other are not be expected. However, it might be hypothesized that a closer stopping criteria on the b*j values might continue to improve the Greedy solution.

All of the algorithms contain some heuristic rules; therefore the answers exhibited in this section are best categorized as near optimal solutions. Both algorithms yield satisfactory answers, but with differences discussed in this Part V and in Part VI.

```
      INITIAL VALUES OF PARAMETERS B
 0.00000010  0.03286799  0.26779460  0.25850234  0.48675927  0.68018747  0.81107932  0.89285344  0.94108090  0.96842135
 0.98345607  0.99151443  0.99573534  0.99789912  0.99898552  0.99951989  0.99977739  0.99989591  0.99995505  0.99998044
 0.99999167  0.99999653  0.99999859  0.99999944  1.00000000

          GREEDY ALGORITHM

          PHASE 1

          INITIAL KIT COMPUTATION

 100  ITEMS IN COMPUTATION

               KIT 5718

  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   3   1   2   1   2   1  16   7   9   9   3   4   1   2  11   1   5   4   6
  1   1   1   3  11

          TARGETS     PERFORMANCE OF 5718 KIT

 COST   OF 5718 KIT=$     4039896.00     NORSGOAL=   4.92074     SDOGOAL=  24.49522

 BUDGET OF INITIAL KIT IS  $      3635906.00

 PRICE(X)=$     3635861.00     E(NORS/X)=   4.84982     E(SDO/X)=  29.47310

          NUMBER OF ITERATIONS   3

   B(J)  0.00000000  0.00172540  0.06441003  0.26115723  0.49893585  0.69276720  0.82435903  0.90472555  0.95038868  0.97499286
 0.98772442  0.99410652  0.99722461  0.99871559  0.99941528  0.99973806  0.99988454  0.99994995  0.99997868  0.99999108
 0.99999634  0.99999852  0.99999942  0.99999977  0.99999991

 WEIGHT :   ~~0.003~~  0.0225

 INITIAL KIT

  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0
  1   0   0   0   1   0   0   0   0   0   3   1   0   1   0   0   1   0   0   1   0   2   2   0   0
  1   1   2   0   1   0   2   0   1   0   0   1   1   1   1   3   2   0   0   1   2   0   1   0   1
  1   0   1   0   1   0   1   2   2   1   1   1   0   1   0   0   0   1   0   1   1   0   1   0   0
  0   1   2   1   2   1   0   4   1   0   1   0   0   0   1   1   1   2   0   1   1   1   1   1   0
  2   1   1   1   0   0   1   1   1   2   1   1   1   1   1   2   1   0   2   1   0   1   0   0   0
  0   0   0   1   0   2   2   0   3   2   1   1  17   6   8   9   4   7   1   3  11   0   6   5   5
  1   2   1   3  12
```

Figure 5.4a Greedy Algorithm Program-A Phase 1 Solution for Kit 5718

INITIAL VALUES OF PARAMETERS B

b_i^o {
0.3 0.3 0.00030001 0.00011449 0.00861967 0.07484124 0.23535679 0.44260779 0.63162239 0.77280201
0.86636736 0.92302780 0.95769561 0.97688367 0.98758667 0.99337292 0.99651327 0.99819555 0.99906609 0.99952462
0.99976077 0.99988181 0.99994153 0.99997163 1.00000000
}

GREEDY ALGORITHM
PHASE 2

TARGETS

NCRSGOAL : 0.13549 SDOGOAL: 163.92674 PRICE=$ 7562583.00

11	1	1	1	1	1	1	1	1	1	1	1	4	3	2	1	2	2	2	1	1	1	1	1	1
2	2	1	4	2	2	1	1	2	4	13	5	1	1	1	3	3	1	3	1	1	7	5	1	1
3	2	7	6	1	3	17	1	2	1	1	1	1	4	2	19	13	4	1	1	15	3	3	1	2
6	3	2	3	8	1	4	4	9	2	4	3	1	15	1	2	1	2	2	3	2	1	1	12	2
2	1	8	4	7	5	2	9	1	5	1	1	1	1	3	1	2	9	1	4	2	2	2	2	2
8	18	9	2	1	1	11	2	2	8	1	13	4	12	9	18	25	1	15	1	1	3	21	2	4
17	6	4	3	2	2	4	1	2	1	2	1	16	7	9	9	3	4	1	2	11	1	5	4	13
1	1	3	3	11	2	1	1	1	1	1	1	1	1	1	1	1	4	5	1	1	1	1	4	15
1	3	3	5	2	1	3	5	5	1	7	2	1	2	1	1	2	5	1	3	5	4	1	1	10
1	19	2	2	2	2	1	4	2	4	2	3	3	3	15	2	2	6	3	2	5	1	4	7	2
2	1	10	4																					

TOTAL NUMBER OF ITEMS UNDER CONSIDERATIONS 264

NUMBER OF ITERATIONS 2

NOTE: Forced Budget = $7,108,451

E(NCRS/X)= 0.03740 E(SDO/X)= 144.56469 PRICE(X)=$ 7108447.00

OPTIMAL KIT

5	0	1	0	0	0	0	0	0	0	0	0	5	3	0	4	1	1	5	2	1	1	1	1	1
2	1	1	2	3	0	2	2	2	3	18	6	1	2	1	0	4	1	3	2	2	10	8	1	1
4	4	10	0	3	2	19	2	5	1	1	2	2	5	4	22	16	2	1	3	16	3	5	0	4
6	2	3	2	8	1	6	7	13	3	6	3	1	16	0	1	2	3	2	4	8	1	2	9	3
1	2	12	5	11	6	2	10	1	2	3	2	2	1	3	3	3	13	1	5	2	3	3	3	2
11	20	11	3	1	2	12	3	3	8	4	15	5	14	10	22	28	0	16	2	1	6	19	3	3
17	6	3	5	2	2	2	1	3	2	1	1	17	6	8	9	4	7	1	3	11	1	6	5	15
1	2	1	3	12	2	0	0	0	0	0	0	0	0	0	1	1	4	6	2	2	0	0	4	17
2	5	4	8	3	3	5	6	6	2	8	3	1	4	3	2	2	4	0	3	5	3	0	0	10
1	11	3	2	5	2	3	6	4	7	3	5	3	4	16	4	4	6	5	2	7	1	6	[19]	1
3	1	10	5																					

b_i^* {
0.30000000 0.30000000 0.30000001 0.39012150 0.00576284 0.08382084 0.25445748 0.46376885 0.64834882 0.78359440
0.87341244 0.92838316 0.96058874 0.97879859 0.98891415 0.99419859 0.99703774 0.99850955 0.99926069 0.99963844
0.99982565 0.99991719 0.99996125 0.99998215 0.99999151
}

WEIGHT : 0.015

Figure 5.4b Greedy Algorithm Program-A Phase 2 Solution: Budget = $7,108,451

```
PRICE OF OPTIMAL AIT =  3672038.00                                  Phase 1

          AIT
 0  0  1  0  0  0  0  0  0  0  0  0  1  1  0  0  0  0  1  1  0  1  0  0  1
 1  0  0  0  1  0  1  1  1  1  1  1  0  1  0  0  1  0  1  1  1  2  2  0  0
 1  1  2  0  2  1  1  1  1  0  0  1  1  1  2  2  2  0  1  2  2  1  2  0  1
 1  1  1  0  2  1  1  2  2  1  1  1  0  1  0  0  1  1  1  1  2  0  1  0  1
 0  2  2  1  2  1  1  1  2  0  1  1  1  0  1  2  1  2  0  1  1  1  1  1  1
 2  2  1  1  1  1  1  1  1  2  2  1  1  1  1  2  1  0  2  2  0  1  0  1  1
 0  1  1  1  1  2  2  0  3  3  1  1  17  6  8  9  4  6  2  3  11  0  5  5  5
 2  2  2  2  12

          dP   4.90458      SJO  26.04894                PHIC=  3672038.00     LAST ITEM ADDED  157     ITEM COST=  41566.00

Probability of n aircraft down

 0.685267931154D-06    0.192501481027D-02    0.500693707119D-04    0.185455216687D+00    0.239311712751D+00
 0.200583377884D+00    0.136792448895D+00    0.628241385133D-01    0.465675699722D-01    0.249044629717D-01
 0.128194417468D-01    0.643585798133D-02    0.312411685988D-02    0.149242578636D-02    0.700022771674D-03
 0.322952900860D-03    0.146503097573D-03    0.654123944599D-04    0.267262038421D-04    0.124001273699D-04
 0.525752476128D-05    0.219787637360D-05    0.893019772352D-06    0.357325305983D-06

 PRICE(AIRS      3672038.00     E(NDRS/AIR=  4.90458     E(SJJ/AIR=  26.04894


 NUMBER OF ITERATIONS IS  15

********** CPU =  2.561 SEC.
```


Figure 5.5a IBM Version of DO29 Program-A Phase 1 Solution for Kit 5718 Budget

```
PRICE OF OPTIMAL KIT =  7007734.00                                  Phase 2

          KIT
  9   0   2   0   0   0   0   0   0   0   0   0   5   3   0   1   1   1   5   2   2   1   1   1   1
  2   1   2   1   3   0   2   2   3   3  17   6   1   2   1   0   4   1   3   2   2  11   8   1   1
  4   4  10   0   1   2  19   3   6   1   1   2   2   5   4  22  16   2   2   4  16   3   5   9   4
  6   2   3   2   7   1   6   7  12   3   6   3   1  16   0   2   3   3   2   4   5   1   2   9   3
  1   4  12   4  11   6   2  10   2   2   3   2   2   1   3   4   3  13   2   3   2   3   4   3   3
 12  20  11   3   2   2  11   3   4   8   4  14   6  15   5  22  28   0  16   4   1   7  19   3   3
 17   5   2   4   2   2   2   3   3   3   1   1  17   6   8   5   4   6   2   3  11   3   5   5  10
  3   2   2   2  12   2   0   0   0   9   9   0   0   9   0   2   1   4   5   2   2   0   9   5  17
  2   5   5   8   4   3   3   7   6   2   3   3   1   4   3   2   1   4   0   3   6   3   1   1  10
  1  11   3   2   3   2   3   6   3   7   3   6   3   4  16   4   4   6   3   2   7   1   7  [10]  2
  3   1  10   3

          VP   8.13031     SOO 145.30523                PRICE  7007734.00     LAST ITEM ADDED  106     ITEM COST=   4000.00
Probability of n aircraft down

  0.0                0.0                0.28351290317270-08    9.67203280350370-04    0.67938181774dD-02
  9.62266991920DD-01 0.16138845928b0+00 0.21145270575b0+00     0.19193617682dD+00     0.14212045152dD+00
  0.93468534714dD-01 0.57093761534fJ-01 0.33d252190077D-01     0.18669927235D0-01     0.10214224289dD-01
  0.54678793756dD-02 0.28730795d4 JJ-02 0.10951b1230320-02     0.75627014006810-03    0.37965651769JD-03
  0.18734340108DD-03 0.91744887257bJ-04 0.44143b9635440-04     0.20925716685JD-04

  PRICE(KITS=   7007734.00     E(NOHS/KIT=   8.13031     E(SOO/KIT= 145.30523

  NUMBER OF ITERATIONS IS  92          NOTE:  Budget (for kit 5718) = $7,562,983

********** CPU = 24.789 SEC.
```

Figure 5.5b IBM Version of D029 Program-A Phase 2 Solution for Kit 5718 Budget

```
INITIAL D029 KIT - COST $ 3665930.12 EXCLUDING NON-OPTIMIZED ITEMS        Phase 1
  LAST ITEM ADDED WAS  157, COST $ 41566.

ITEM      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
QUANTITY  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0

ITEM     26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
QUANTITY  1  0  0  0  1  0  1  0  0  0  3  1  0  1  0  0  2  0  1  2  0  4  3  0  0

ITEM     51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75
QUANTITY  1  3  2  0  2  0  4  1  3  0  0  1  1  1  1  3  3  0  0  3  3  1  1  0  1

ITEM     76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
QUANTITY  2  0  1  0  1  0  1  2  2  1  2  1  0  1  0  0  1  1  1  1  2  0  1  0  1

ITEM    101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125
QUANTITY  0   1   2   1   3   1   1   3   1   0   2   0   1   0   1   2   1   2   0   1   1   1   1   3   1

ITEM    126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
QUANTITY  3   1   1   1   0   1   1   1   2   1   1   3   2   1   2   1   0   2   1   0   3   0   1   1

ITEM    151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
QUANTITY  0   1   0   1   0   2   2   0   3   3   1   0  17   7   8   9   4   6   2   2  11   0   5   5   5

ITEM    176 177 178 179 180
QUANTITY  1   2   1   2  12

 CALCULATION OF EXPECTED NMCS AND SDO BY PROGRAM
 THE EXPECTED STOCK DUE OUTS ARE  29.48304986954
 THE EXPECTED NMCS ARE            4.856314621866

 CALCULATION OF EXPECTED NMCS AND SDO BY ACCURATE FORMULA

 PROBABILITY OF N AIRCRAFT DOWN - FORMAT (N, PROB)
  0 4.892433028117E-10     1 .0011893455154816      2 .0529225917111 5      3 .1946731766295        4 .2487002483633
  5 .2004929067431         6 .1320544424227         7 .0785162575406 6      8 .0439167317028 3      9 .0235230362703 1
 10 .0121835260962 1      11 .0061375861753 05     12 .0030187661842 37    13 .0014535727180 81    14 .0006884855567676
 15 .0003183773221096     16 .0001450952893443     17 .0000649897811015 9  18 .0000286048698088398 19 .0000123667786851
 20 .0000052487352206 75  21 .0000021856530700 86  22 6.924823227119E-7    23 3.571986972872E-7    24 2.257284979619E-7
    17.083 SECONDS - FINISH CALC. ENMCS AND ESDO

 EXPECTED VALUES
  STOCK DUE OUTS          28.5643894861 88
  NMCS AIRCRAFT           4.8562384164531
  STD. DEV. OF NMCS       1.900014897499
  NEW ENMCS               1.409170265463
  AVAILABILITY            .6983074494534
  NO. OF UNAVAIL. AC      16.84062121312

ERR ITEM ADDED TO KIT: ITEM 158, COST  $ 41566.
ERR ITEM ADDED TO KIT: ITEM 162, COST  $ 13000.
ERR ITEM ADDED TO KIT: ITEM 172, COST  $ 29011.
PASS 1 DONE - 17.205 SECONDS
```

Figure 5.6a D029-CDC Program-A Phase 1 Solution for Kit 5718 Budget

```
OPTIMAL D029 KIT - COST = 7108651.15 EXCLUDING NON-OPTIMIZED ITEMS      Phase 2
  LAST ITEM ADDED WAS  108, COST = 4000.

ITEM      1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25
QUANTITY  9   0   1   0   0   0   0   0   0   0   0   0   5   2   0   1   1   1   5   2   1   1   1   1   1

ITEM     26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50
QUANTITY  2   1   2   1   3   0   3   2   2   3  17   7   2   2   1   0   4   3   3   3   2  13   9   1   1

ITEM     51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75
QUANTITY  5   6  10   0   3   2  20   2   7   1   1   2   2   5   4  22  16   2   2   3  17   3   5   0   5

ITEM     76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
QUANTITY  7   2   3   2   7   1   6   7  12   3   6   3   1  16   0   2   2   4   2   4   5   2   2   9   3

ITEM    101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125
QUANTITY  2   2  13   5  11   6   2  10   1   2   3   2   2   1   3   3   3  12   1   5   3   3   3   3   2

ITEM    126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
QUANTITY 11  20  10   3   1   2  11   3   3   8   5  14   5  13   9  22  28   0  16   2   1   6  19   4   4

ITEM    151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
QUANTITY 17   5   2   4   2   2   2   1   3   3   1   1  17   7   6   9   4   6   2   2  11   1   5   5  10

ITEM    176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
QUANTITY  1   2   1   2  12   2   0   0   0   0   0   0   0   0   0   1   1   5   5   2   2   0   0   5  17

ITEM    201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225
QUANTITY  2   5   4   8   3   3   5   6   6   2   8   3   1   4   3   3   2   4   0   3   5   4   0   0  10

ITEM    226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250
QUANTITY  1  11   3   2   6   2   3   6   5   7   3   5   3   4  16   5   4   6   5   2   8   2   9   1   1

ITEM    251 252 253 254
QUANTITY  4   2  10   5


CALCULATION OF EXPECTED NMCS AND SDO BY PROGRAM
THE EXPECTED STOCK DUE OUTS ARE  152.6630849836
THE EXPECTED NMCS ARE            8.127570556009

CALCULATION OF EXPECTED NMCS AND SDO BY ACCURATE FORMULA

PROBABILITY OF N AIRCRAFT DOWN - FORMAT (N, PROB)
 0 3.2050807799196E-45     1 1.0279787627654E-1M     2 4.000105281146-4      3 .000068330710413 27     4 .00685747965940 5
 5 .06262492200893         6 .161M057354181          7 .211432988213 4       8 .191624270958 1         9 .14183 87959723
10 .09329668409167        11 .05703461354094        12 .03320061440932      13 .01866766663789        14 .01021564693069
15 .00546879113064 6      16 .00287374758193        17 .00148549221441 4    18 .00075041961352 7      19 .00037972118888 16
20 .00014797609160 26     21 .00009175689189789     22 .00004414902814176   23 .00002092812767884     24 .00001700545405 48
   60.143 SECONDS - FINISH CALC. ENMCS AND ESDO

EXPECTED VALUES
 STOCK DUE OUTS          149.6567302179  <==
 NMCS AIRCRAFT           8.127350742485  <==
 STD. DEV. OF NMCS       2.160065998357
 NEW ENMCS               3.026570632156
 AVAILABILITY            .9916849635273 16
 NO. OF UNAVAIL. AC      23.95994487534

                                  Budget (is for kit 5718) = $7,562,983 ($7,563,001.17 originally)
```

Figure 5.6b D029 CDC Program-A Phase 2 Solution for Kit 5718 Budget

```
        INITIAL VALUES OF PARAMETERS B
b_j^o [0.00000016  0.00286799  0.00779460  0.25850234  0.48975927  0.66818747  0.81107932  0.89285344  0.94108090  0.98842135
       0.98345667  0.99151443  0.99573534  0.99789912  0.99894554  0.99951595  0.99977739  0.99989691  0.99995505  0.99998044
       0.99999167  0.99999653  0.99999859  0.99999944  1.00000000

              GREEDY ALGORITHM

              PHASE 1


              INITIAL KIT COMPUTATION

   180  ITEMS IN COMPUTATION

                    KIT 5718

     1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
     1   1   1   1   1   1   3   1   2   1   2   1  16   7   9   9   3   4   1   2  11   1   5   4   6
     1   1   1   3  11

              TARGETS     PERFORMANCE OF 5718 KIT

   COST   OF 5718 KIT=$      4039896.00     NONSCORAL=   4.92074     SCORGOAL=  24.49522

   BUDGET OF INITIAL KIT IS  $       3635906.00

   PRICE(KIT=$      3635861.00     E(NORS/KIT=   4.84982     E(SCO/KIT=  29.47310


              NUMBER OF ITERATIONS   3

b_j^* [ B(J)  0.00000000  0.30172540  0.06441003  0.16115723  0.49893585  0.69276720  0.82435903  0.90472555  0.95038808  0.97499286
       0.98772442  0.99410652  0.99722461  0.99871559  0.99941528  0.99973806  0.99988454  0.99994995  0.99997868  0.99999108
       0.99999634  0.99999852  0.99999942  0.99999977  0.99999991

   WEIGHT =      0.023


   INITIAL KIT

     0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0
     1   0   0   0   1   0   0   0   0   0   3   1   0   1   0   0   1   0   0   1   0   2   2   0   0
     1   1   2   0   1   0   2   0   1   0   0   1   1   1   1   2   2   0   0   1   2   0   1   0   1
     1   0   1   0   1   0   1   2   2   1   1   1   0   1   0   0   0   1   0   1   1   0   1   0   0
     0   1   2   1   2   1   0   1   1   0   1   1   0   0   1   1   1   1   2   0   1   1   1   1   0
     2   1   1   1   0   0   1   1   1   2   1   1   1   1   1   1   2   1   0   2   1   0   1   0   0
     0   0   0   1   0   2   2   0   3   2   1   1  17   6   8   9   4   7   1   3  11   0   6   5   5
     1   2   1   3  12
```

Figure 5.7a Phase 1 Greedy Algorithm Solution: Budget of Kit 5718

INITIAL VALUES OF PARAMETERS B

b_j^o =

0.0	0.0	0.00000001	0.00011449	0.00861567	0.07484124	0.23535679	0.44260775	0.63182230	0.77280201
0.86636330	0.92392780	0.95769561	0.97683397	0.98754667	0.99337298	0.99651327	0.99818555	0.99906608	0.99952462
0.99978077	0.99988191	0.99994153	0.99997163	1.00000000					

GREEDY ALGORITHM
PHASE 2

TARGETS

NORSGOAL = 8.13540 SODGOAL = 163.32674 PRICE= $ 7562983.00

11	1	1	1	1	1	1	1	1	1	1	1	4	3	2	1	2	2	2	1	1	1	1	1	1
2	2	1	4	2	2	1	1	2	4	13	5	1	1	1	3	3	1	3	1	1	1	7	5	1	1
3	2	7	6	1	3	17	1	2	1	1	1	1	4	2	19	13	4	1	1	15	3	5	1	2
6	3	2	3	8	1	4	4	9	2	4	3	1	15	1	2	1	2	2	3	2	1	1	12	2
2	1	8	4	7	5	2	9	1	5	1	1	1	1	3	1	2	5	1	4	2	2	2	2	2
8	18	9	2	1	1	11	2	2	5	1	13	4	12	8	18	25	1	15	1	1	3	21	2	4
17	6	4	3	2	2	4	1	2	1	2	1	16	7	9	9	3	4	1	2	11	1	5	4	13
1	1	3	3	11	2	1	1	1	1	1	1	1	1	1	1	1	4	5	1	1	1	1	4	15
1	3	3	5	2	1	3	5	5	1	7	2	1	2	1	1	2	5	1	3	5	4	1	1	10
1	10	2	2	2	2	1	4	2	4	2	3	2	3	15	2	2	6	3	2	5	1	4	7	2
2	1	10	4																					

TOTAL NUMBER OF ITEMS UNDER CONSIDERATIONS :54

NUMBER OF ITERATIONS 4

NOTE: Budget (is for kit 5718) = $7,562,983

ENORS/KIT= 7.15853 ESOD/KIT= 36.81477 PRICE/KIT= $ 7562944.00

OPTIMAL KIT

10	0	2	0	0	0	0	1	0	0	0	0	0	3	1	1	2	2	5	2	2	2	2	2	2
3	2	1	3	4	1	2	2	2	4	19	7	1	3	2	2	5	2	4	3	2	16	9	1	1
4	4	19	0	3	3	20	2	5	1	2	3	2	6	5	24	17	4	1	3	18	4	6	1	4
7	3	3	3	9	2	7	8	14	4	7	4	2	17	1	2	2	4	1	5	6	2	3	11	3
2	2	13	5	11	7	3	11	1	3	3	2	2	2	4	3	4	14	2	6	3	3	3	3	3
12	21	12	3	1	2	13	3	3	5	4	16	6	15	11	23	29	1	16	2	1	6	20	3	4
18	7	4	5	2	2	2	1	3	2	1	1	17	6	8	9	4	7	1	3	11	1	6	5	11
1	2	2	3	12	3	0	0	1	0	0	0	1	1	1	1	1	5	7	2	1	0	1	5	19
2	5	5	9	3	3	6	7	7	2	9	4	2	4	3	2	2	5	0	4	6	4	0	1	11
2	12	4	3	5	2	3	7	4	7	4	6	4	5	17	4	5	7	5	3	8	2	7	11	2
3	1	11	6																					

B

b_j^* =

0.00000000	0.00000000	0.00000935	0.00376217	0.05835153	0.22116612	0.43810253	0.63279580	0.77581119	0.88953345
0.92667958	0.95987318	0.97850019	0.98868473	0.99413854	0.99700804	0.99849417	0.99925267	0.99963432	0.99982364
0.99991621	0.99996080	0.99998195	0.99999192	0.99999636					

WEIGHT = 0.020

Figure 5.7b Phase 2 Greedy Algorithm Solution: Budget of $7,562,983

```
      INITIAL VALUES OF PARAMETERS B
b_j^o { 0.00000010  0.00286799  0.08779460  0.25830234  0.48975927  0.68018747  0.81107932  0.89265344  0.94108090  0.96642135
        0.98345607  0.99151443  0.99573534  0.99789912  0.99896552  0.99951589  0.99977739  0.99989891  0.99995595  0.99998044
        0.99999167  0.99999653  0.99999859  0.99999944  1.00000000

              GREEDY ALGORITHM

              PHASE 1

              INITIAL KIT COMPUTATION

    180  ITEMS IN COMPUTATION

                   KIT 5718

     1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
     1  1  1  1  1  1  1  3  1  2  1  2  1 16  7  9  9  3  4  1  2 11  1  5  4  6
     1  1  1  3 11

              TARGETS      PERFORMANCE OF 5718 KIT

    COST   OF 5718 KIT= $      4039896.00      NORSGOAL=   4.92074      SDUGOAL=  24.49522

    BUDGET OF INITIAL KIT IS  $       3635906.00

    PRICE(X)=$      3635881.00      E(NORS/X)=   4.84982      E(SDO/X)=  29.47210

              NUMBER OF ITERATIONS   3

b_j^* { B(J)  0.00000000  0.00172548  0.08641063  0.26115723  0.49893585  0.69276720  0.82435903  0.90472555  0.95038808  0.97499266
      { 0.98772442  0.99410652  0.99722461  0.99871559  0.99941528  0.99973606  0.99988454  0.99994995  0.99997868  0.99999108
      { 0.99999634  0.99999852  0.99999942  0.99999977  0.99999991

    WEIGHT =    0.6225

    INITIAL KIT

     0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0
     1  0  0  0  1  0  0  0  0  0  0  1  0  1  0  0  0  1  0  0  1  0  2  2  0  0
     1  1  2  0  1  0  2  0  1  0  0  1  1  1  1  3  2  0  0  1  2  0  1  0  1
     1  0  1  0  1  0  1  2  2  1  1  1  0  1  0  0  0  1  0  1  1  0  1  0  0
     0  1  2  1  2  1  0  4  1  0  1  0  0  0  1  1  1  2  0  1  1  1  1  1  0
     2  1  1  1  0  0  1  1  1  2  1  1  1  1  1  2  1  1  2  0  1  1  1  1  1  0
     0  0  0  1  0  2  2  0  3  2  1  1 17  6  8  9  4  7  1  3 11  0  6  5  5
     1  2  1  3 12
```

Figure 5.8a Phase 1 Greedy Algorithm Solution: Budget of Kit 5718

INITIAL VALUES OF PARAMETERS B

b_j^0 =

0.0	0.0	0.00000001	0.00011449	0.00861567	0.07484124	0.23535679	0.44269779	0.63162230	0.77280201
0.88636330	0.92352780	0.95769561	0.97686397	0.98754667	0.99337298	0.99651327	0.99818555	0.99906048	0.99952462
0.99975077	0.99988101	0.99994153	0.99997163	1.00000000					

GREEDY ALGORITHM
PHASE 2

TARGETS

NORSGOAL = 8.13540 SDOGUAL= 163.92674 PRICE=$ 7562963.00

11	1	1	1	1	1	1	1	1	1	1	1	4	3	2	1	2	2	2	1	1	1	1	1	1
2	2	1	4	2	2	1	1	2	4	13	5	1	1	1	3	3	1	3	1	1	7	5	1	1
3	2	7	6	1	3	17	1	2	1	1	1	1	4	2	19	13	4	1	1	15	3	3	1	2
6	3	2	3	8	1	4	4	9	2	4	3	1	15	1	2	1	2	2	3	2	1	1	12	2
2	1	8	4	7	5	2	9	1	5	1	1	1	1	3	1	2	5	1	4	2	2	2	2	2
8	18	9	2	1	1	11	2	2	5	1	13	4	12	8	18	25	1	15	1	1	3	21	2	4
17	6	4	3	2	2	4	1	2	1	2	1	16	7	9	9	3	4	1	2	11	1	5	4	13
1	1	3	3	11	2	1	1	1	1	1	1	1	1	1	1	1	4	5	1	1	1	1	4	15
1	3	5	5	2	1	3	5	5	1	7	2	1	2	1	1	2	5	1	3	5	4	1	1	10
1	10	2	2	2	2	1	4	2	4	2	3	2	3	15	2	2	6	3	2	5	1	4	7	2
2	1	10	4																					

TOTAL NUMBER OF ITEEMS UNDER CONSIDERATIONS 254

NUMBER OF ITERATIONS 2

NOTE: Forced Budget = $7,075,000

E(NORS/XJ)= 8.17363 E(SDO/XJ)= 145.09709 PRICE(XJ)=$ 7074907.00

OPTIMAL KIT

9	0	1	0	0	0	0	0	0	0	0	0	5	3	0	1	1	1	5	2	1	1	1	1	1
2	1	1	2	4	0	2	2	2	3	18	6	1	2	1	0	4	1	3	2	2	10	8	1	1
4	4	10	0	3	2	19	2	5	1	1	2	2	5	4	22	16	2	1	3	16	3	6	0	4
6	2	3	2	8	1	6	7	13	3	6	3	1	16	0	2	2	3	2	4	5	1	2	9	3
1	2	12	5	11	6	2	10	1	2	3	2	2	1	3	3	3	13	1	5	2	3	3	3	2
11	20	11	3	1	2	12	3	3	8	4	15	5	14	10	22	28	6	16	2	1	6	18	3	3
17	6	3	4	2	2	2	1	5	2	1	1	17	6	8	9	4	7	1	3	11	1	6	5	10
1	2	1	3	12	2	0	0	0	0	0	0	0	0	0	1	1	4	6	2	2	0	0	4	17
2	5	4	8	3	3	5	6	6	2	8	3	1	4	3	2	2	4	0	3	5	3	0	0	10
1	11	3	2	5	2	3	6	4	7	3	5	2	4	16	4	4	6	5	2	7	1	6	10	1
3	1	10	5																					

B

b_j^* =

0.00000000	0.00000000	0.00000001	0.00010537	0.00873863	0.07697626	0.23847377	0.44173677	0.62555852	0.76410186
0.85783866	0.91707057	0.95281514	0.97368799	0.98557714	0.99221554	0.99565975	0.99782983	0.99887524	0.99943004
0.99971477	0.99985963	0.99993211	0.99996775	0.99998496					

WEIGHT = 0.01501

Figure 5.8b Phase 2 Greedy Algorithm Solution: Budget of $7,075,000

PART VI

SUMMARY OF RESULTS

This research provides a new and more general model, the "multiobjective model," for the WRSK problem. This multiobjective approach resulted in an improved measure of supply performance which relates E(NORS/X) and E(SDO/X) for a WRSK kit in a meaningful way.

Three potential solution algorithms for this new "multiobjective model" were investigated. The Saddle Point Algorithm was the most general, but suffered from the duality gap. The Primal Search Algorithm made special use of the properties of the objective function, but lacked flexibility to adjust the search direction in an effective way.

A third method proposed by R. B. S. Brooks in [3] and subsequently modified during this research lead to the development of the "Greedy Algorithm" as the solution procedure. This algorithm requires that the objective function be approximated by a separable function. This research has accomplished the required separability of the multiobjective model. This allows the use of marginal analysis in a manner that achieves the highest rate of return per unit of performance measure per dollar invested in the kit as items are added to the kit.

Computational experimentation has shown that the Greedy Algorithm is faster than the other two potential algorithms. An important result of this research has been that the Greedy Algorithm program is faster and provides a better solution than the current DO29 using F4D data.

The Greedy Algorithm has several advantages. It requires only one ratio of marginal return be updated whenever one unit of an item is added

to a kit. This is an immediate consequence of having achieved a separable approximation of the objective function of the new multiobjective model. This contributes to the ability of the Greedy Algorithm to yield an improved kit compared to the DO29-CDC kit, and in shorter time when properly compared to the DO29-IBM program.

A disadvantage of the Greedy Algorithm is that it is constructed on the convergence of the b_j parameters. The generation of a new set of parameter values from a previous set takes one full iteration. The parameter values of this new set are compared to the previous set. Such iterations and comparisons are repeated until the parameters are sufficiently close, at which time the algorithm terminates. Therefore, the last iteration is used only to conclude that the optimal solution is as good as the next to the last solution. Thus there is always an extra iteration that does not improve the kit configuration. Another disadvantage might be that kits for intermediate budgets might be desired, but must await determination of the optimal b_j parameters. Also the weighting factor w must be determined in advance, and additional experimentation may be required to determine the appropriate value.

Another product of this research has been the modification of the current DO29 program for compatibility with IBM FORTRAN compilers. This effort has provided some insight into the problems with the current DO29 program. The word packing and unpacking required by the CDC compilers, in order to save memory, consumes a great deal of time. The storing of some probability values, and the interpolation of others, when needed, does not appear to be a good policy. Interpolation underestimates the probabilities; hence, the marginal rates of return are less accurate and lead to kit solutions not as good as those obtained by the Greedy Algorithm.

The storing of probabilities, and interpolation of others, does appear to possibly contribute to errors in a kit configuration. The Greedy Algorithm and the DO29-IBM program produces $X^*_{249} = 10$, but the DO29-CDC program produces $X^*_{249} = 1$. The source of this error should be understood and eliminated from the current DO29-CDC program.

Comparison of kit results, indicates that the kits obtained by the Greedy Algorithm or the DO29 program are similar. Approximately 75% of the kit items are present in the same amounts and the remainings deviate only by 1 or 2. This is confirmation of the recognition that the current DO29 model is acceptable. The Greedy Algorithm appears to offer the potential of substantial time savings, and a slightly better kit configuration.

REFERENCES

1. Bellman, R. E. and S. E. Dreyfus, Applied Dynamic Programming, Princeton, New Jersey, Princeton University Press, 1962.

2. Brooks, R. B. S. and A. Geoffrion, "Finding Everett's Lagrange Multipliers by Linear Programming," Operations Research, Vol. 14, 1966 , pp. 1149-1152.

3. Brooks, R. B. S., C. A. Gillen, and J. Y. Lu, "Alternative Measures of Supply Performance," The RAND Corporation, RM-6094-PR, August, 1969.

4. Fox, B., "Discrete Optimization Via Marginal Analysis," Management Science, Vol. 13, 1966, pp. 210-216.

5. Garfinkel, R. S. and G. L Nemhauser, Integer Programming, New York, New York: John Wiley and Sons, Inc., 1972.

6. Miller, B. L., "Unconstrained Optimization in the Integers," RAND Corporation, RM-6165-PR, January, 1970.

7. Sherbrooke, C. C., "Metric: A Multi-Echelon Technique for Recoverable Item Control," Operations Research, Vol. 16, 1968, pp. 122-141.

8. W.P.A.F.L.C., Preliminary Evaluation of DO29 WRSK Model, Technical Report, 1980.

9. Yuceer, U., "Optimization of Single Resource Allocation Problems with Nonseparable Objective Functions," Ph.D. dissertation, Oklahoma State University, December, 1980.

APPENDIX A

A.1 Computer Code for the Primal Search Algorithm

A.2 Computer Code for the Modified Algorithm of Brooks, et al.

A.3 Computer Code for the Saddle Point Search Algorithm

APPENDIX A.1

COMPUTER CODE FOR THE

PRIMAL SEARCH

ALGORITHM

A List of the Variables in the Computer Program

ALFA	Stepsize
BASLA	The Subroutine that finds an initial point
BETA	Product of the parameters p (user determined) and p_k
BUTCE	Budget
C(100)	Cost of item I
CBETA	The parameter p_k
CDOTD	The dot product of the cost and direction vectors
CDOTX	The dot product of the cost vector and the current solution, also the cost of the kit
CKARE	The dot product the cost vector with itself
COST	Cost of the current solution
DENOM	The dot product of the Cost vector the vector of first differences at a solution
DIF	The first difference
DIREC	The direction vector
ENORS	E(NORS/X)
EPS	Parameter for testing the sufficiently closeness
IOX	Integer Optimal Solution
ITETA	Indicator of the search direction. If ITETA = 1, the search is in the positive direction, if ITETA = -1, it is in the negative direction.
IX(100)	Current integer solution
IY(100)	Initial solution
KODE	Indicator of the result of the search, "1" means success, "0" means failure
N	Total number of items in the problem
NEWP	New integer point
NITER	Number of iterations already performed

NNORS	Number of NORS function evaluations
NSTEP (NITER)	Number of steps within iteration NITER
M	Upper bound
PAR	Parameter p (user determined)
PDFOP	Function routine which calculates the individual poisson terms
POISON (40,100)	The matrix which contains the cumulative Poisson sums
SIRALA (X,IR)	The subroutine which ranks the array X in descending order. IR(k) gives the rank of X(k).
TABLE (POISON)	The subroutine which fills in the matrix POISON
UCOS	Cost of NEWP
UNORS	E(NORS/NEWP)
VNORS	Function Routine to calculate E(NORS/X) funtion
XLAG	Lagrange parameter (user determined)
XLAM(I)	Poisson demand rate of item I.

FORTRAN Listing of the Computer Program

```
$JOB TIME=05
C ************************************
C *                                  *
C *                                  *
C *    A PRIMAL SEARCH ALGORITHM     *
C *                                  *
C *                                  *
C ************************************
C
C   MAIN PROGRAM: TO READ IN THE DATA
C                 TO WRITE OUT THE SOLUTIONS
C
      DIMENSION IX(100),IOX(100),IY(100),DELTA(100)
      COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE,PAR/BLOCK4.
     *NITER,NSTEP(500)/BLOCK5/POISON(40,10))/COUNT/NNORS
      EPS=.00005
C   INPUT THE NECESSARY INFORMATION
C
      NNORS=0
      READ(5,90) NOPR
      READ(5,100) N,M,BUTCE
      READ(5,110) (C(I),I=1,N)
      READ(5,110) (XLAM(I),I=1,N)
      READ(5,120) PAR,XLAG
C
C   CALCULATE THE PROBABILITIES
C
      CALL TABLE(POISON)
C
C   FIND AN INITIAL SOLUTION
      CALL BASLA(XLAG,IY)
      DO 10 I=1,N
   10 IX(I)=IY(I)
C
C   START THE OPTIMIZATION
C
      CALL OPTIM(IX,IOX,ENORS,COST)
C
C   PRINT THE RESULTS
C
      WRITE(6,400)
      WRITE(6,405) NOPR
      WRITE(6,209) N,BUTCE
      WRITE(6,200)
      DO 20 I=1,N
   20 WRITE(6,210) I,C(I),XLAM(I),IY(I),IOX(I)
      WRITE(6,201) ENORS,COST
      WRITE(6,205) NNORS
      WRITE(6,220) NITER,(NSTEP(I),I=1,NITER)
      WRITE(6,250) PAR,XLAG
   90 FORMAT(I4)
  100 FORMAT(2I4,F10.0)
  110 FORMAT(10F8.0)
  120 FORMAT(2F10.0)
  200 FORMAT(///,38X,'INITIAL',5X,'OPTIMAL',/,5X,'ITEM',3X,'COST/UNIT',6
     1X,'DEMAND',4X,'SOLUTION',4X,'SOLUTION',//)
  201 FORMAT(///,5X,10HE(NORS/X)=,F10.5,5X,9HCOST(X)=$,F10.2)
  205 FORMAT(//,5X,'# OF NORS EVALUATIONS=',I4)
  209 FORMAT(///,5X,11H# OF ITEMS=,I4,3X,8HBUDGET=$,F10.2)
  210 FORMAT(5X,'#',I3,2(2X,F10.2),2(8X,I4))
```

```
      220 FORMAT(//,5X,'# OF ITERATIONS=',I4,/,5X,'# OF STEPS IN EACH ITERA
     *TION:',(5X,2014))
      250 FORMAT(//,5X,'LAMBDA=',F10.2,5X,'LAGRANGE MULTIPLIER=',F10.8,/)
      400 FORMAT(1H1,//,10X,'A PRIMAL SEARCH METHOD')
      405 FORMAT(///,5X,'PROBLEM NUMBER',I4)
          STOP
          END

          SUBROUTINE OPTIM(IX,IOX,ENORS,COST)
C *                                   *
C *    OPTIMIZATION ROUTINE           *
C *                                   *
          DIMENSION IX(100),IOX(100),DIREC(100),DELTA(100),NEWP(100),
         1XNEW(100),NEIGH(100),IY(100),IRANK(100),XDEL(100)
          COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE,PAR/BLOCK4
         *NITER,NSTEP(500)
C
          NITER=1
          ENORS=VNORS(IX)
          COST=PROD(IX)
          CKARE=DOTPR(C)
C
C   EVALUATE THE FIRST DIFFERENCES
      200 CALL TJREV(IX,DELTA)
          NSTEP(NITER)=0
          DENOM=DOTPR(DELTA)
          CBETA=DENOM/CKARE
C
C   EVALUATE THE DIRECTION PARAMETER PK
          BETA=PAR*CBETA
C
C   DETERMINE THE SEARCH DIRECTION
          DO 10 I=1,N
          DIREC(I)=DELTA(I)-BETA*C(I)
          WSA=ABS(DIREC(I))
          IF(I.EQ.1) GO TO 15
          IF(WSA.LE.XSI) GO TO 10
       15 XSI=WSA
       10 CONTINUE
C
C   NORMALIZE THE DIRECTION
          DO 11 I=1,N
       11 DIREC(I)=DIREC(I)/XSI
C   CALCULATE THE STEP SIZE ALPHA
          CDOTD=DOTPR(DIREC)
          CDOTX=PROD(IX)
          XN=SQRT(FLOAT(N))
          AMAX=(BUTCE-CDOTX)/(CDOTD*XN)
          IF(AMAX.LT.1.0) AMAX=1.0
          MMAX=IFIX(AMAX+.5)
          AMAX=FLOAT(MMAX)
          ALFA=AMAX
C
C   OBTAIN A NEW POINT
      100 DO 20 I=1,N
          WN=FLOAT(IX(I))+ALFA*DIREC(I)+.5
          IF(WN.GE.1.) GO TO 20
          WN=1.0
       20 NEWP(I)=IFIX(WN)
          NSTEP(NITER)=NSTEP(NITER)+1
```

```
      C   CALCULATE ITS COST $ E(NORS/X)
   75         UCOS=PROD(NEWP)
   76         UNORS=VNORS(NEWP)
      C   IF INFEASIBLE , PERFORM A SEARCH IN THE NEGATIVE DIRECTION
   77         IF(UCOS.GT.BUTCE) GO TO 30
      C   IF A BETTER FEASIBLE POINT, REPEAT THE STEPS
   78         IF(UNORS.LT.ENORS) GO TO 500
   79         ITETA=1
   80         GO TO 40
   81      30 ITETA=-1
   82      40 CALL SEARCH(ITETA,UCOS,UNORS,ENORS,NEWP,KODE)
   83         IF(KODE.EQ.1) GO TO 500
   84         IF(ALFA.EQ.1.) GO TO 900
   85         ALFA=ALFA-1.
   86         GO TO 100
   87     900 UCOS=COST
   88         UNORS=ENORS
   89         CALL SEARCH(1,UCOS,UNORS,ENORS,IX,KODE)
   90         IF(KODE) 600,1000,600
   91     500 DO 45 I=1,N
   92      45 IX(I)=NEWP(I)
   93     600 COST=UCOS
   94         ENORS=UNORS
   95         NITER=NITER+1
   96         GO TO 200
   97    1000 DO 50 I=1,N
   98      50 IOX(I)=IX(I)
   99         RETURN
  100         END

  101         SUBROUTINE SEARCH(ITETA,UCOS,UNORS,ENORS,NEWP,KODE)
      C
      C   THIS SUBROUTINE PERFORMS SEARCHES EITHER IN POSITIVE OR NEGATIVE DI
      C
  102         DIMENSION NEWP(100),NEIGH(100),IRANK(100),DELTA(100),INSET(100),D
             *L(100)
  103         COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE,PAR/BLOCK4
             *NITER,NSTEP(500)
      C
  104         NSTEP(NITER)=NSTEP(NITER)+1
  105         IF(ITETA.EQ.1) GO TO 9
      C
      C   CALCULATE THE FIRST DIFFERENCES FOR NEGATIVE DIRECTION
  106         CALL TURNO(NEWP,DELTA)
  107         GO TO 11
      C   CALCULATE THE FIRST DIFFERENCES
  108       9 CALL TUREV(NEWP,DELTA)
  109      11 DO 10 I=1,N
  110         DELTA(I)=FLOAT(ITETA)*DELTA(I)
  111      10 DEL(I)=DELTA(I)
      C
      C   RANK THE FIRST DIFFERENCES
  112         CALL SIRALA(DEL,IRANK)
  113         DO 20 I=1,N
  114         DO 15 J=1,N
  115         JV=J
  116         IF(IRANK(J).EQ.I) GO TO 17
  117      15 CONTINUE
  118      17 INSET(I)=JV
  119      20 CONTINUE
```

```
      C
      C   CHECK WHETHER THERE IS A BETTER FEASIBLE POINT
      C
  120       DO 25 I=1,N
  121       JVAL=INSET(I)
  122       COSJ=UCOS+FLOAT(ITETA)*C(JVAL)
  123       IF(COSJ.GT.BUTCE) GO TO 25
  124       DO 35 JJ=1,N
  125    35 NEIGH(JJ)=NEWP(JJ)
  126       IF(NEWP(JVAL).EQ.0.AND.ITETA.EQ.-1) GO TO 25
  127       NEIGH(JVAL)=NEWP(JVAL)+ITETA
  128       XNORS=UNORS-DELTA(JVAL)
  129       IF(XNORS-ENORS) 100,120,110
  130    25 CONTINUE
  131       GO TO 120
  132   110 DO 40 I=1,N
  133    40 NEWP(I)=NEIGH(I)
  134       UNORS=XNORS
  135       UCOS=COSJ
      C   IF NO BETTER FEASIBLE POINT, FAILURE $ KODE=0
  136   120 KODE=0
  137       RETURN
      C
      C   IF A BETTER FEASIBLE POINT, SUCCESS $ KODE=1
  138   100 KODE=1
  139       DO 70 I=1,N
  140    70 NEWP(I)=NEIGH(I)
  141       UNORS=XNORS
  142       UCOS=COSJ
  143       RETURN
  144       END

  145       FUNCTION PROD(IX)
      C
      C   TO CALCULATE THE DOT PRODUCT C.IX
      C
  146       DIMENSION IX(100)
      C
  147       COMMON N,M/BLOCK1/C(100)
  148       SUM=0.0
  149       DO 10 I=1,N
  150       IXI=IX(I)
  151    10 SUM=SUM+FLOAT(IXI)*C(I)
  152       PROD=SUM
  153       RETURN
  154       END

  155       FUNCTION DOTPR(X)
      C
      C   TO CALCULATE THE DOT PRODUCT C.X
      C
  156       DIMENSION X(100)
  157       COMMON N,M/BLOCK1/C(100)
  158       SUM=0.0
  159       DO 10 J=1,N
  160       SUM=SUM+C(J)*X(J)
  161    10 CONTINUE
  162       DOTPR=SUM
  163       RETURN
  164       END
```

```
165           FUNCTION VNORS(IX)
      C
      C    TO EVALUATE THE E(NORS/X)
      C
166           DIMENSION IX(100),PM(100,20),PRO(20)
167           COMMON N,M/BLOCKS/POISON(40,100)/COUNT/NNORS
168           DOUBLE PRECISION SUM,PRO,TOPL,PM,XW
      C
169           EPS=.00001
170           EPSQ=.00000001
171           NNORS=NNORS+1
172           M1=M+1
173           SUM=0.D+00
174           DO 20 JJ=1,M1
175           PRO(JJ)=1.D+00
176           DO 26 II=1,N
177           IXJ=IX(II)+JJ-1
178           IF(IXJ.GT.0) GO TO 24
179           XX=POFOP(IXJ,II)
180           GO TO 25
181        24 IF(IXJ.GT.40) IXJ=40
182           XX=POISON(IXJ,II)
183        25 IF(XX.LT.EPSQ) XX=0.0
184           IF(PRO(JJ).LT.EPSQ) GO TO 20
185           PRO(JJ)=PRO(JJ)*XX
186        26 CONTINUE
187        20 SUM=SUM+1.-PRO(JJ)
188           TOPL=0.D+00
189           DO 30 I=1,N
190           K=0
191        40 MXK=IX(I)+M+2+K
192           WX=POFOP(MXK,I)
193           WW=FLOAT(K+1)*WX
194           IF(WX.LT.EPS) GO TO 30
195           TOPL=TOPL+WW
196           K=K+1
197           GO TO 40
198        30 CONTINUE
199           VNORS=SUM+TOPL
200           RETURN
201           END

202           SUBROUTINE TUREV(IX,DELTA)
      C
      C    TO EVALUATE THE FIRST DIFFERENCES
      C
203           DIMENSION IX(100),DELTA(100)
204           COMMON N,M
      C
205           DO 10 K=1,N
206           DELTA(K)=DIF(IX,K)
207        10 CONTINUE
208           RETURN
209           END

210           SUBROUTINE TURND(NEWP,DELTA)
      C
      C    TO EVALUATE THE FIRST DIFFERENCES   IN NEGATIVE DIRECTION
      C
211           DIMENSION NEWP(100),NEIGH(100),DELTA(100)
```

```
212           COMMON N,M
      C
213           DO 10 I=1,N
214           DO 20 J=1,N
215           IF(J.EQ.I) GO TO 25
216           NEIGH(J)=NEWP(J)
217           GO TO 20
218        25 NEIGH(J)=NEWP(J)-1
219           IF(NEIGH(J).LT.0) NEIGH(J)=0
220        20 CONTINUE
221           DELTA(I)=DIF(NEIGH,I)
222        10 CONTINUE
223           RETURN
224           END

225           FUNCTION DIF(IX,K)
      C
      C    TO COMPUTE THE FIRST DIFFERENCE OF ITEM K
      C
226           DIMENSION IX(100)
227           COMMON N,M/BLOCK5/POISON(40,100)
228           DOUBLE PRECISION PRO,SUM,EPSQ
      C
229           EPSQ=1.D-08
230           M1=M+1
231           SUM=0.D+00
232           DO 20 J1=1,M1
233           PRO=1.D+00
234           J=J1-1
235           DO 30 I=1,N
236           IXJ=IX(I)+J
237           IF(I.EQ.K) GO TO 25
238           IF(IXJ.EQ.0) GO TO 23
239           IF(IXJ.GT.39) IXJ=39
240           PRO=PRO*POISON(IXJ,I)
241           GO TO 30
242        23 IXJ1=IXJ
243           GO TO 26
244        25 IXJ1=IXJ+1
245        26 IF(PRO.LT.EPSQ) GO TO 20
246           PRO=PRO*POFOP(IXJ1,I)
247        30 CONTINUE
248        20 SUM=SUM+PRO
249           IXM=IX(K)+M+1
250           IF(IXM.GT.40) IXM=40
251           DIF=SUM+1.-POISON(IXM,K)
252           RETURN
253           END

254           SUBROUTINE TABLE(POISON)
      C
      C    TO FILL IN THE ARRAY FOR CUMULATIVE POISSON SUMS
      C
255           DIMENSION POISON(40,100)
256           COMMON N,M/BLOCK2/XLAM(100)
257           DOUBLE PRECISION EPS,TERM,SUMX
      C
258           EPS=1.D-06
259           DO 10 J=1,N
260           TERM=1.D+00
```

```
261           SUMX=1.D+00
262           DO 20 I=1,40
263           TERM=TERM*XLAM(J)/FLOAT(I)
264           IX=I
265           IF(TERM.LT.EPS) GO TO 25
266           SUMX=SUMX+TERM
267           POISON(I,J)=EXP(-XLAM(J))*SUMX
268           IF(POISON(I,J).GT.1.0) POISON(I,J)=1.0
269        20 CONTINUE
270        25 IF(IX.EQ.40) GO TO 10
271           DO 30 I=IX,40
272        30 POISON(I,J)=1.0
273        10 CONTINUE
274           RETURN
275           END

276           FUNCTION PDFOP(K,I)
      C
      C    TO CALCULATE INDIVIDJAL TERMS OF THE POISSON DISTRIBUTION
      C
277           COMMON /BLOCK2/XLAM(100)
278           DOUBLE PRECISION SUM,WW
      C
279           SUM=0.D+00
280           IF(K.LT.2) GO TO 10
281           DO 15 J=2,K
282           SUM=SUM+ALOG(FLOAT(J))
283        15 CONTINUE
284        10 WW=-XLAM(I)+K*ALOG(XLAM(I))-SUM
285           IF(WW.LT.-15.0) GO TO 20
286           PDFOP=DEXP(WW)
287           GO TO 25
288        20 PDFOP=0.0
289        25 RETURN
290           END

291           SUBROUTINE SIRALA(X,IR)
      C
      C    THIS SUBROUTINE RANKS AN ARRAY OF ELEMENTS IN DESCENDING ORDER
      C    IR(K) IS THE RANK OF ITEM K IN THIS ORDERING
      C
292           COMMON N,M
293           DIMENSION X(100),IR(100),ISEQ(100),Y(100)
      C
294           DO 9 I=1,N
295        9  ISEQ(I)=I
296           DO 10 I=1,N
297           XMIN=X(1)
298           IMINJ=1
299           IMIN=ISEQ(1)
300           N1=N-I+1
301           DO 20 J=1,N1
302           IF(X(J).GT.XMIN) GO TO 21
303           GO TO 20
304        21 XMIN=X(J)
305           IMIN=ISEQ(J)
306           IMINJ=J
307        20 CONTINUE
308           IR(IMIN)=I
309           IF(N1.EQ.1) GO TO 10
```

```
310           N2=N1-1
311           DO 30 J=1,N2
312           IF(J.LT.IMINJ) GO TO 33
313           J1=J+1
314           ISEQ(J)=ISEQ(J1)
315           Y(J)=X(J1)
316           GO TO 29
317        33 Y(J)=X(J)
318        29 X(J)=Y(J)
319        30 CONTINUE
320        10 CONTINUE
321           RETURN
322           END

323           SUBROUTINE BASLA(XLAG,IX)
          C
          C    THIS SUBROUTIN FINDS AN INITIAL POINT WITH A PREDETERMINED
          C    LAGRANGE PARAMETER
          C
324           DIMENSION IX(100)
325           COMMON N,M/BLOCK1/C(100)/BLOCK5/POISON(40,100)
          C
326           DO 10 I=1,N
327           IW=0
328           A=XLAG*C(I)+PDFOP(IW,I)-1.
329           IF(A.GE.0.0) GO TO 10
330        15 IW=IW+1
331           B=A+PDFOP(IW,I)
332           IF(A.LE.0.0.AND.B.GT.0.0) GO TO 10
333           A=B
334           GO TO 15
335        10 IX(I)=IW
336           RETURN
337           END

          $ENTRY
```

A PRIMAL SEARCH METHOD

PROBLEM NUMBER 1

OF ITEMS= 5 BUDGET=$ 25000.00

ITEM	COST/UNIT	DEMAND	INITIAL SOLUTION	OPTIMAL SOLUTION
# 1	2980.00	2.10	2	3
# 2	1751.00	1.50	2	2
# 3	462.00	1.20	3	3
# 4	1500.00	5.00	6	6
# 5	345.00	3.50	6	6

E(NORS/X)= 0.98571 COST(X)=$ 24898.00

OF NORS EVALUATIONS= 3

OF ITERATIONS= 2
OF STEPS IN EACH ITERATION: 2 3

LAMBDA= 0.05 LAGRANGE MULTIPLIER=0.00020000

STATEMENTS EXECUTED= 13876

CORE USAGE OBJECT CODE= 15256 BYTES, ARRAY AREA= 43028 BYTES, TOTAL AREA AVAILABLE= 145408 BYTES

DIAGNOSTICS NUMBER OF ERRORS= 0, NUMBER OF WARNINGS= 0, NUMBER OF EXTENSIONS= 0

COMPILE TIME= 0.27 SEC, EXECUTION TIME= 0.24 SEC, 12.35.52 MONDAY 4 AUG 80 WATFIV - JUN 1977

APPENDIX A.2

COMPUTER CODE FOR THE MODIFIED ALGORITHM OF BROOKS ET AL.

A List of the Variables in the Computer Program

BO	Coefficients $b_0, b_1, \ldots, b_m$
C(I)	Cost of item I
COST	Cost of the current solution X
ENORS	E(NORS/X)
EPS	Parameter for testing sufficiently closeness
F(100,50)	Matrix for the cumulative poisson sums
ICOUN	Counter for the number of iterations
IMAX	The index I for which RATIO(I) is maximum
IX(100)	Current solution
M	An upper bound
N	The total number of items under consideration
NOPR	Problem number
NSTEP (ICOUN)	Number of steps in iteration ICOUN
OPTIM	The optimization routine
PDFOP	Function routine which calculates the individual poisson terms
RATIO (100)	Ratio of return per dollar investment
RMAX	Ratio (IMAX)
TABLE	The subroutine to fill in the matrix F(100,50)
XLAM (100)	Poisson demand rates
XINVT	Budget
VNORS	Function routine to calculate E(NORS/X)

The FORTRAN Listing of the Computer Program

```
      C
      C ***********************************************
      C
      C   MODIFIED SOLUTION METHOD OF BROOKS ET AL
      C
      C ***********************************************
      C
 1          DIMENSION IX(100),BO(100)
 2          COMMON N,M/BLOCK1/C(100),XINVT/BLOCK2/XLAM(100)/BLOCK4/F(100,50)/
     *LOCK5/ICOUN,NSTEP(50)
      C
 3          EPS=.001
 4          EPSQ=EPS**2
      C
      C   INPUT THE DATA
      C
 5          READ(5,90) NOPR
 6          READ(5,100) N,M,XINVT
 7          READ(5,110) (C(I),I=1,N)
 8          READ(5,110) (XLAM(I),I=1,N)
 9          WRITE(6,400)
10          WRITE(6,405) NOPR
11          WRITE(6,105) N,XINVT
12          M1=M+1
      C
      C   INITIALIZE THE PARAMETERS B
      C
13          DO 10 J=1,M
14       10 BO(J)=0.0
15          BO(M1)=1.0
      C
      C   CALCULATE THE PROBABILITIES
16          CALL TABLE(F)
17          ICOUN=1
      C
      C   PERFORM THE OPTIMIZATION
      C
18    1000  CALL OPTIM(BO,IX,COST)
      C
      C   CHECK IF B-NEW IS SUFFICIENTLY CLOSE TO B-OLD
      C
19          SUMSQ=0.0
20          DO 20 J1=1,M1
21          PRO=1.0
22          DO 40 I5=1,N
23          IXJ=IX(I5)+J1
24          IF(IXJ.GT.50) GO TO 40
25          PRO=PRO*F(I5,IXJ)
26       40 CONTINUE
27          IF(PRO.LT.EPSQ) PRO=0.0
28          SUMSQ=SUMSQ+((BO(J1)-PRO)**2)
29          BO(J1)=PRO
30       20 CONTINUE
      C
      C   IF SO, TERMINATE AND PRINT THE RESULTS
      C
31          IF(SUMSQ.LT.EPS) GO TO 500
      C   OTHERWISE REPEAT THE STEPS
32          ICOUN=ICOUN+1
```

```
33          GO TO 1000
34      500 WRITE(6,240)
35          DO 30 I=1,N
36       30 WRITE(6,250) I,C(I),XLAM(I),IX(I)
37          ENORS=VNORS(IX)
38          WRITE(6,201) ENORS,COST
39          WRITE(6,300) (BO(J),J=1,M1)
40          WRITE(6,301) ICOUN,(NSTEP(I),I=1,ICOUN)
41       90 FORMAT(I4)
42      100 FORMAT(2I4,F10.0)
43      105 FORMAT(///,5X,' N=',I4,5X,' BUDGET=$',F10.2)
44      110 FORMAT(10F8.0)
45      201 FORMAT(//,5X,10HE(NORS/X)=,F10.5,2X,9HCOST(X)=$,F10.2)
46      240 FORMAT(///,38X,'OPTIMAL',/,5X,'ITEM',3X,'COST/UNIT',5X,'DEMAND',4X
           1,'SOLUTION',//)
47      250 FORMAT(5X,'#',I3,2(2X,F10.2),8X,I4)
48      300 FORMAT(//,5X,'B:',10F10.3)
49      301 FORMAT(///,5X,16H# OF ITERATIONS=,I4,//,(5X,'# OF STEPS IN EACH I
           *TERATION:',2014))
50      400 FORMAT(1H1,//,5X,'THE MODIFIED SOLUTION METHOD OF BROOKS ET AL',/,
           *)
51      405 FORMAT(///,5X,'PROBLEM NUMBER',I4)
52          STOP
53          END

54          SUBROUTINE OPTIM(BO,IX,COST)
      C
      C   OPTIMIZATION ROUTINE PERFORMS THE MARGINAL ANALYSIS
      C
55          DIMENSION IX(100),BO(100),RATIO(100),MR(100)
56          COMMON N,M/BLOCK1/C(100),XINVT/BLOCK3/F(100,5)/BLOCK5/ICOUN,NSTEP
           *(50)
      C
57          M1=M+1
58          COST=0.0
59          DO 10 I=1,N
60          IX(I)=0
      C
      C   CALCULATE THE RATIO OF RETURN PER EACH DOLLAR INVESTED FOR EACH ITE
61          SUM=0.0
62          DO 20 J=1,M1
63          JJ=J+1
64          WW=F(I,JJ)/F(I,J)
65       20 SUM=SUM+BO(J)*ALOG(WW)
66       10 RATIO(I)=SUM/C(I)
67          NSTEP(ICOUN)=1
68      200 NOR=0
69          NSTEP(ICOUN)=NSTEP(ICOUN)+1
70          DO 21 I=1,N
71       21 MR(I)=-1
72      100 IMR=0
      C
      C   FIND THE MAXIMUM RATIO & ADD ONE UNIT OF THIS ITEM TO THE KIT
      C
73          DO 20 I=1,N
74          IF(MR(I).EQ.1) GO TO 25
75          IMR=IMR+1
76          IF(IMR.EQ.1) GO TO 26
77          IF(RATIO(I).LE.RMAX) GO TO 25
78       26 RMAX=RATIO(I)
```

```
      IMAX=I
   25 CONTINUE
      MR(IMAX)=1
C
C   CALCULATE THE COST OF THE NEW KIT
      XCOST=COST+C(IMAX)
C
C   IF COST EXCEEDS THE BUDGET, FIND THE ITEM WHICH WILL YIELD MAXIMUM
C   WITHOUT EXCEEDING THE BUDGET, THEN RETURN THE MAIN PROG.
C
      IF(XCOST.GT.XINVT) GO TO 30
      COST=XCOST
      IX(IMAX)=IX(IMAX)+1
      SUM=0.0
      DO 35 J=1,M1
      IXJ=IX(IMAX)+J
      IXJ1=IXJ+1
      IF(IXJ.GE.50) GO TO 35
      WW=F(IMAX,IXJ1)/F(IMAX,IXJ)
      SUM=SUM+BC(J)*ALOG(WW)
   35 CONTINUE
C   OTHERWISE UPDATE THE RATIO(IMAX), REPEAT THE STEPS
      RATIO(IMAX)=SUM/C(IMAX)
      GO TO 200
   30 NOR=NOR+1
      IF(NOR.LT.N) GO TO 100
      RETURN
      END

      SUBROUTINE TABLE(F)
C
C   CALCULATING CUMULATIVE POISSON SUMS
C
      DIMENSION F(100,50)
      COMMON N,M/BLOCK1/C(100),XINVT/BLOCK2/XLAM(100)
      DOUBLE PRECISION EPS,TERM,SUM
C
      EPS=1.D-06
      DO 10 I=1,N
      F(I,1)=POFOP(0,I)
      TERM=1.D+00
      SUM=1.D+00
      DO 30 J=2,50
      TERM=TERM*XLAM(I)/FLOAT(J-1)
      JX=J
      IF(TERM.LT.EPS) GO TO 25
      SUM=SUM+TERM
      F(I,J)=EXP(-XLAM(I))*SUM
      IF(F(I,J).GT.1.0) F(I,J)=1.0
   30 CONTINUE
   25 IF(JX.EQ.50) GO TO 10
      DO 35 J=JX,50
   35 F(I,J)=1.0
   10 CONTINUE
      RETURN
      END

      FUNCTION POFOP(K,I)
C
C   CALCULATING THE INDIVIDUAL POISSON TERMS
```

```
      C
  124         COMMON /BLOCK2/XLAM(100)
  125         DOUBLE PRECISION SUM,WW
      C
  126         SUM=0.D+00
  127         IF(K.LT.2) GO TO 10
  128         DO 15 J=2,K
  129         SUM=SUM+ALOG(FLOAT(J))
  130      15 CONTINUE
  131      10 WW=-XLAM(I)+K*ALOG(XLAM(I))-SUM
  132         IF(WW.LT.-15.D) GO TO 20
  133         POFOP=DEXP(WW)
  134         GO TO 25
  135      20 POFOP=0.0
  136      25 RETURN
  137         END

  138         FUNCTION VNORS(IX)
      C
      C   CALCULATING THE E(NORS/X) FUNCTION
      C
  139         DIMENSION IX(100),PRO(20)
  140         COMMON N,M/BLOCK4/F(100,50)
  141         DOUBLE PRECISION SUM,PRO,TOPL
      C
  142         EPS=.00001
  143         M1=M+1
  144         SUM=0.D+00
  145         DO 10 J=1,M1
  146         PRO(J)=1.D+00
  147         DO 20 I=1,N
  148         IXJ=IX(I)+J
  149      20 PRO(J)=PRO(J)*F(I,IXJ)
  150      10 SUM=SUM+1.-PRO(J)
  151         TOPL=0.D+00
  152         DO 30 I=1,N
  153         K=0
  154      40 MXK=IX(I)+M+2+K
  155         WX=POFOP(MXK,I)
  156         WW=FLOAT(K+1)*WX
  157         IF(WX.LT.EPS) GO TO 30
  158         TOPL=TOPL+WW
  159         K=K+1
  160         GO TO 40
  161      30 CONTINUE
  162         VNORS=SUM+TOPL
  163         RETURN
  164         END

      $ENTRY
```

THE MODIFIED SOLUTION METHOD OF BROOKS ET AL

PROBLEM NUMBER 1

N= 5 BUDGET=$ 25000.00

ITEM	COST/UNIT	DEMAND	OPTIMAL SOLUTION
# 1	2980.00	2.10	2
# 2	1751.00	1.50	2
# 3	462.00	1.20	4
# 4	1500.00	5.00	7
# 5	345.00	3.50	9

EINCRS/X)= 0.98619 COST(X)= $ 24915.00

B: 0.450 0.728 0.891 0.962 0.988 0.996

OF ITERATIONS= 3

OF STEPS IN EACH ITERATION: 23 26 26

STATEMENTS EXECUTED= 8963

CORE USAGE OBJECT CODE= 7320 BYTES, ARRAY AREA= 22800 BYTES, TOTAL AREA AVAILABLE= 145408 BYTES

DIAGNOSTICS NUMBER OF ERRORS= 0, NUMBER OF WARNINGS= 0, NUMBER OF EXTENSIONS= 0

COMPILE TIME= 0.13 SEC, EXECUTION TIME= 0.14 SEC, 12.11.57 MONDAY 4 AUG 80 WATFIV - JUN 1977

APPENDIX A.3

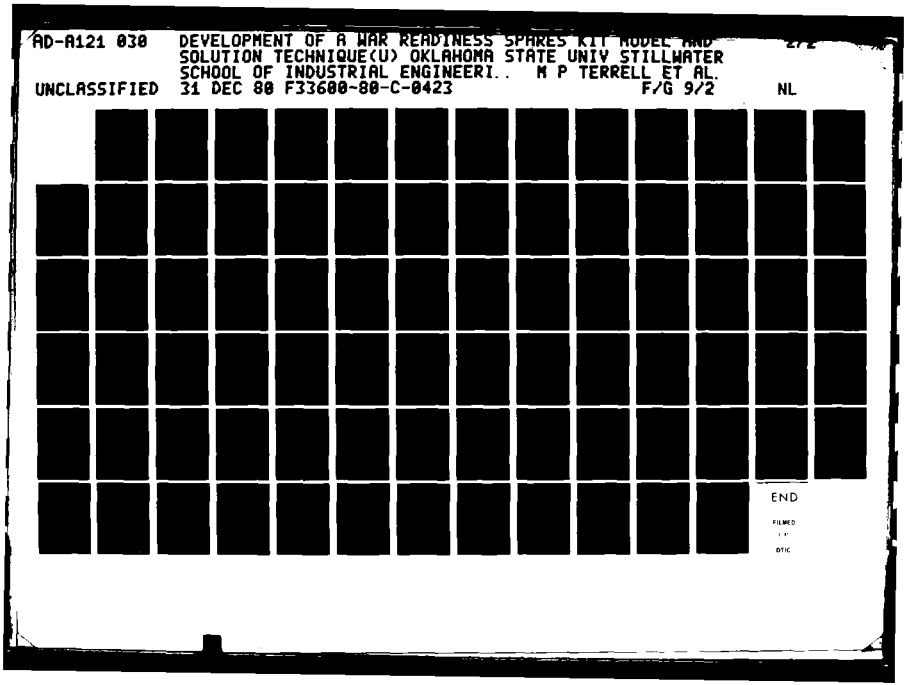
COMPUTER CODE FOR THE SADDLE POINT SEARCH ALGORITHM

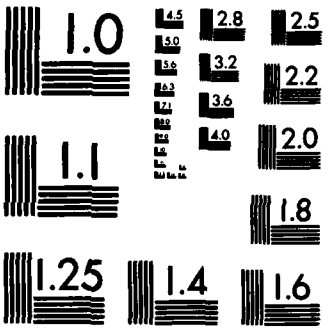
A List of the Variables in the Computer Program

A(50)	The cost of the current solution for subroutine DUALPR
AXM	Minimum of A(I) which exceed BUTCE; A(IXM)
AXP	Maximum of A(I) which are less than BUTCE; A(IXP)
BASLA	Subroutine which finds an initial point
BUTCE	Budget
C(100)	Cost of the item
COST	Total cost of the current solution
CX(50)	E(NORS/X) (used only in DUALPR)
CXM	CX(IXM)
CXP	CX(IXP)
DIF	Function routine for calculating first differences
DOTPR	The dot product of two vectors
DUALPR	Subroutine which solves the dual problem to obtain the Lagrange multiplier
ENORS	E(NORS/X)
INPUT	The subroutine which reads in the necessary data and initializes the arrays
ISTAR	Starting solution X^O
IX(100)	Current solution
IX0(100)	Solution obtained in the Subroutine BASLA
IX1(100)	Solution obtained in the Subroutine SEARCH
IX2(100)	Solution obtained in the Subroutine TARAMA
1X0(100)	Final and/or optimal solution
IY(100)	Initial solution
KPAR	Parameter for finding an initial point in BASLA
M	An upper bound

N	Total number of items under consideration
NARA	Number of times Subroutine ARAMA called
NBAS	Number of time Subroutine BASLA called
NITER	Number of iterations
NOPG	Number of points generated by the Saddle Point Search Algorithm
NOPR	Problem number
NPA1	Number of times Part 1 called
NPA2	Number of times Part 2 called
NPA3	Number of times Part 3 called
NSEA	Number of times SEARCH called
NVNOR	Number of E(NORS/X) function evaluations
OPTIM	Optimization Routine
PART1	The subroutine which performs a test by calculating an upper bound to the linear program in determining whether a component can be increased by one
PART2	The subroutine which calculates an upper bound for the sum of first differences
PART3	The subroutine which performs the same test as in PART1 but sets up the linear program explicitly
PDFOP	Function routine for calculating individual poisson terms
PLAG	Current Lagrange multiplier
POISON	Matrix for the cumulative poisson terms
POLD	Previous Lagrange multiplier
SEARCH	The subroutine that makes a one dimensional search for each component separately
SIRALA (X,IR)	The subroutine that ranks the array X in descending order
TABLE	The subroutine which fills in the matrix POISON

TARAMA	The subroutine which makes an attempt for improvement heuristically
VNORS	The function routine to calculate E(NORS/X) + λ CX
VXO	E(NORS/IXO)
VX1	E(NORS/IX1)
VX2	E(NORS/IX2)
XLAM(100)	Poisson demand rates
XLPRO	The Linear Programming Routine which solves problems of the type max CX subject to $AX \leq b$ $X \geq 0$.


AD-A121 030 DEVELOPMENT OF A WAR READINESS SPARES KIT MODEL AND
SOLUTION TECHNIQUE(U) OKLAHOMA STATE UNIV STILLWATER
SCHOOL OF INDUSTRIAL ENGINEERI.. M P TERRELL ET AL.
UNCLASSIFIED 31 DEC 80 F33600-80-C-0423 F/G 9/2 NL
END
FILMED
DTIC



FORTRAN Listing of the Computer Program

```
      $JOB TIME=05
      C ************************************
      C *                                  *
      C *                                  *
      C *  A  SADDLE POINT SEARCH ALGORITHM *
      C *                                  *
      C *                                  *
      C ************************************
      C
      C   MAIN PROGRAM
      C
    1       DIMENSION IX(100),IY(100)
      C
      C   INPUT THE DATA
    2       CALL INPUT(NOPR,KPAR,PLAG,IY)
      C
      C   PERFORM THE OPTIMIZATION
      C
    3       CALL OPTIM(PLAG,KPAR,IY,ENORS,IX,COST)
      C
      C   PRINT THE RESULTS
      C
    4       CALL OTPUT(PLAG,NOPR,IY,ENORS,IX,COST)
      C
    5       STOP
    6       END

    7       SUBROUTINE INPUT(NOPR,KPAR,PLAG,IY)
      C
      C   TO READ IN THE DATA
      C
    8       DIMENSION IY(100)
    9       COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE
   10       COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
           *NPA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
      C
   11       READ(5,90) NOPR
   12       READ(5,100) N,M,BUTCE
   13       READ(5,110) (C(I),I=1,N)
   14       READ(5,110) (XLAM(I),I=1,N)
   15       READ(5,150) KPAR,ISTAR,PLAG
      C
      C   INITIALIZE THE COUNTERS
   16       NITER=1
   17       NVNOR=0
   18       DO 19 I=1,50
   19       NBAS(I)=0
   20       NSEA(I)=0
   21       NTAR(I)=0
   22       NARA(I)=0
   23       NPA1(I)=0
   24       NPA2(I)=0
   25       NPA3(I)=0
   26    19 CONTINUE
   27       DO 10 I=1,N
   28    10 IY(I)=ISTAR
      C
      C   CALCULATE THE CUMULATIVE POISSON SUMS
      C
   29       CALL TABLE(POISON)
```

```
      90 FORMAT(I4)
     100 FORMAT(2I4,F10.0)
     110 FORMAT(10F8.0)
     150 FORMAT(2I4,F10.0)
         RETURN
         END

         SUBROUTINE OTPUT(PLAG,NOPR,IY,ENORS,IX,COST)
C
C    WRITING OUT THE RESULTS
C
         DIMENSION IY(100),IX(100)
         COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE
         COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
        *NPA2(50),NPA3(50),NVNOR/TAB/POISON(5),100
C
         WRITE(6,300)
         WRITE(6,405) NOPR
         WRITE(6,205) N,BUTCE
         WRITE(6,220)
         DO 20 I=1,N
      20 WRITE(6,225) I,C(I),XLAM(I),IY(I),IX(I)
         WRITE(6,210) ENORS,COST
         WRITE(6,230) PLAG
         WRITE(6,240) NITER,NVNOR
         WRITE(6,245)
         DO 30 I=1,NITER
      30 WRITE(6,250) I,NBAS(I),NSEA(I),NTAR(I),NARA(I),NPA1(I),NPA2(I),NP/
        1A3(I)
     205 FORMAT(5X,'NUMBER OF ITEMS=',I4,2X,'BUDGET=$',F10.2)
     220 FORMAT(///,38X,'INITIAL',5X,'OPTIMAL',/,5X,'ITEM',3X,'COST/UNIT',6
        1X,'DEMAND',4X,'SOLUTION',4X,'SOLUTION',//)
     225 FORMAT(5X,'#',I3,2(2X,F10.2),2(8X,I4))
     210 FORMAT(//,5X,'E(NORS/X)=',F10.5,5X,'COST(X)=$',F10.2)
     230 FORMAT(/,5X,'LAGRANGE MULTIPLIER=',F10.6)
     240 FORMAT(//,5X,'# OF ITERATIONS=',I4,//,5X,'# OF NORS EVALUATIONS=',
        1I4)
     245 FORMAT(//,5X,'# OF  STEPS IN EACH ITERATION',//,5X,'ITERATION #',4
        *X,'STEP1  STEP2  STEP3  STEP4  PART1  PART2  PART3',/)
     250 FORMAT(12X,I4,2X,7(3X,I4))
     300 FORMAT(1H1,//,5X,'THE SADDLE POINT SEARCH ALGORITHM',//)
     405 FORMAT(///,5X,'PROBLEM NUMBER',I4,/)
         RETURN
         END

         SUBROUTINE OPTIM(PLAG,KPAR,IYO,VXA,IXO,COST)
C
C    OPTIMIZATION ROUTINE
C
         DIMENSION IXO(100),IX1(100),IXO(100),IYO(100),CX(50),A(50),IX2(100
        *)
         COMMON N,M/BLOCK3/BUTCE/BLOCK4/NOPG/BLOCK1/C(100)
         COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
        *NPA2(50),NPA3(50),NVNOR
C
         EPS=.000002
         NOPG=1
         ZZ=0.0
         CX(NOPG)=VNORS(ZZ,IYO)
         A(NOPG)=DOTPR(IYO)
```

```
      C
      C   STEP1: FIND AN INITIAL POINT
      C
   73    500 CALL BASLA(PLAG,KPAR,IX0)
   74        IF(NITER.GT.1) GO TO 20
   75        DO 25 I=1,N
   76     25 IX1(I)=IX0(I)
      C
      C    STEP2: FIND A STATIONARY POINT
      C
   77     20 CALL SEARCH(PLAG,IX0,IX1)
   78        VX1=VNORS(PLAG,IX1)
      C
      C   STEP3: A HEURISTIC IMPROVEMENT TEST
      C
   79        CALL TARAMA(PLAG,IX1,VX1,IX2,VX2)
      C
      C   COMPARE THE NEW POINT WITH THE INCUMBENT POINT
   80        IF(VX2-VX1) 14,12,12
   81     14 DO 13 I=1,N
   82     13 IX0(I)=IX2(I)
   83        GO TO 20
      C
      C   STEP4: REDUCTION OF POTENTIAL CANDIDATES BY CONTRADICTION
      C
   84     12 CALL ARAMA(PLAG,IX2,VX2,IX0,VX0)
   85        IF(VX0-VX1) 10,15,15
   86     10 DO 11 I=1,N
   87     11 IX0(I)=IX0(I)
   88        GO TO 20
   89     15 NOPG=NOPG+1
      C
      C   READJUST THE COEFFICEINT OF THE LINEAR PROGRAM
      C
   90        A(NOPG)=DOTPR(IX0)
   91        COST=A(NOPG)
   92        VXA=VNORS(ZZ,IX0)
   93        CX(NOPG)=VXA
   94        POLD=PLAG
      C
      C   SOLVE THE DUAL LP TO OBTAIN THE LAGRANGE MULTIPLIER
      C
   95        CALL DUALPR(CX,A,PLAG)
      C
      C   IF THE NEW MULTIPLIER IS SUFFICIENTLY CLOSE TO OLD ONE, TERMINATE
      C   OTHERWISE REPEAT THE ITERATIONS
      C
   96        PARK=ABS(POLD-PLAG)
   97        IF(PARK.LT.EPS) GO TO 1000
   98        NITER=NITER+1
   99        GO TO 500
  100 1000 RETURN
  101        END

  102        SUBROUTINE DUALPR(CX,A,X)
      C
      C   SOLVE THE DUAL TO OBTAIN A LAGRANGE MULTIPLIER
      C
  103        DIMENSION CX(50),A(50)
  104        COMMON /BLOCK3/BUICE/BLOCK4/NOPG
```

```
105           IX=0
106           IY=0
107           DO 10 I=1,NOPG
108           IF(A(I)-BUICE) 20,20,30
109        20 IY=IY+1
110           IF(IY.EQ.1) GO TO 25
111           IF(A(I)-AXP) 10,10,25
112        25 AXP=A(I)
113           CXP=CX(I)
114           GO TO 10
115        30 IX=IX+1
116           IF(IX.EQ.1) GO TO 35
117           IF(A(I)-AXM) 35,10,10
118        35 AXM=A(I)
119           CXM=CX(I)
120        10 CONTINUE
121           X=(CXP-CXM)/(AXM-AXP)
122           RETURN
123           END

124           SUBROUTINE BASLA(XLAG,KPAR,IX)
      C
      C    FINDS AN INITIAL POINT
      C
125           DIMENSION IX(100)
126           COMMON N,M/BLOCK1/C(100)
127           COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
             *NPA2(50),NPA3(50),NVNOR
128           NBAS(NITER)=NBAS(NITER)+1
129           DO 10 I=1,N
130           IW=0
131           A=XLAG*C(I)+PDFOP(IW,I)-1.
132           IF(A.GE.0.0) GO TO 11
133        15 IW=IW+1
134           B=A+PDFOP(IW,I)
135           IF(A.LE.0.0.AND.B.GT.0.0) GO TO 11
136           A=B
137           GO TO 15
138        11 IW=IW-KPAR
139           IF(IW.LT.0) IW=0
140        10 IX(I)=IW
141           RETURN
142           END

143           SUBROUTINE SEARCH(XLAG,IX,IXO)
      C
      C    PERFORMS ONE DIMENSIONAL SEARCH FOR EACH COMPONENT SEPARATELY
      C
144           DIMENSION IX(100),IXO(100),IY(100)
145           COMMON N,M
146           COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
             *NPA2(50),NPA3(50),NVNOR
      C
147           NSEA(NITER)=NSEA(NITER)+1
148           DO 10 I=1,N
      C
      C    CALCULATE THE FIRST DIFFERENCE
149           X1=DIF(XLAG,IX,I)
      C
```

```
C   IF IT IS POSITIVE , THE POINT X(I) IS STATIONARY, OTHERWISE
C   FIND X(I) SUCH THAT ITS FIRST DIFFERENCE IS POSITIVE
C
150       IF(X1) 20,25,25
151    20 DO 11 J=1,N
152       IF(I.EQ.J) GO TO 12
153       IY(J)=IX(J)
154       GO TO 11
155    12 IY(J)=IX(J)+1
156    11 CONTINUE
157       Y1=DIF(XLAG,IY,I)
158       IF(Y1) 30,36,36
159    30 DO 13 J=1,N
160       IF(I-J) 13,14,13
161    14 IX(J)=IX(J)+1
162    13 CONTINUE
163       X1=Y1
164       GO TO 20
165    25 DO 15 J=1,N
166       IF(I.EQ.J) GO TO 16
167       IY(J)=IX(J)
168       GO TO 15
169    16 IY(J)=IX(J)-1
170       IF(IY(J).LT.0) GO TO 36
171    15 CONTINUE
172       Y2=DIF(XLAG,IY,I)
173       IF(Y2) 36,36,40
174    40 DO 60 J=1,N
175    60 IX(J)=IY(J)
176       X1=Y2
177       GO TO 25
178    36 DO 65 J=1,N
179    65 IXO(J)=IX(J)
180    10 CONTINUE
181       RETURN
182       END

183       SUBROUTINE TARAMA(XLAG,IX,VX,IXO,VXO)
C
C   OBTAINS IMPROVEMENT  HEURISTICALLY
C
184       DIMENSION IX(100),IY(100),RATIO(100),IRANK(100),IXO(100)
185       COMMON N,M
186       COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
         *NPA2(50),NPA3(50),NVNJR
C
187       NTAR(NITER)=NTAR(NITER)+1
188       ITETA=1
189   100 DO 10 I=1,N
190       DO 20 J=1,N
191       IF(J.EQ.I) GO TO 25
192       IF(ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 25
193       IY(J)=IX(J)+ITETA
194       GO TO 20
195    25 IY(J)=IX(J)
196    20 CONTINUE
197       ALFB=DIF(XLAG,IY,I)
198       ALFA=DIF(XLAG,IX,I)
199       RATIO(I)=ALFB/ALFA
200    10 CONTINUE
```

```
      C
      C   RANK THE RATIO OF FIRST DIFFERENCES OF EACH COMPONENT
      C
201           CALL SIRALA(RATIO, IRANK)
202           DO 30 I=1,N
203           DO 35 J=1,N
204           IF(IRANK(J).GT.I) GO TO 38
205           IF(ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 38
206           IY(J)=IX(J)+ITETA
207           GO TO 35
208        38 IY(J)=IX(J)
209        35 CONTINUE
      C
      C   CALCULATE ITS E(NORS/X)
210           VY=VNORS(XLAG, IY)
      C
      C   COMPARE WITH THE INCUMBENT SOLUTION
211           IF(VY.LT.VX) GO TO 50
212        30 CONTINUE
213           IF(ITETA.EQ.-1) GO TO 90
214           ITETA=-1
215           GO TO 100
216        50 DO 60 I=1,N
217        60 IXO(I)=IY(I)
218           VXO=VY
219           GO TO 110
220        90 DO 70 I=1,N
221        70 IXO(I)=IX(I)
222           VXO=VX
223       110 RETURN
224           END

225           SUBROUTINE ARAMA(XLAG,IX,VX,IXO,VXO)
      C
      C   REDUCES THE SET OF POTENTIAL CANDIDATES BY CONTRADICTION
      C
226           DIMENSION IX(100),IXO(100),IY(100),IZ(100),ISIR(100),JARR(100),MO
             *C(100)
227           COMMON N,M/BLOCK5/ITETA/BLOCK8/NOZE,JARR/BLOCK6/WLAM
228           COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
             *NPA2(50),NPA3(50),NVNOR
229           NARA(NITER)=NARA(NITER)+1
      C
230           WLAM=XLAG
231           ITETA=1
232       500 ICOUN=0
233           NOZE=0
234           IKODE=0
235       400 DO 10 I=1,N
236           IF(NOZE.EQ.0) GO TO 19
237           DO 15 II=1,NOZE
238           IF(JARR(II).EQ.I) GO TO 10
239        15 CONTINUE
240        19 DO 20 J=1,N
241           IF(J.EQ.I) GO TO 25
242           IF(ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 25
243           IF(NOZE.EQ.0) GO TO 29
244           DO 26 JJ=1,NOZE
245           IF(JARR(JJ).EQ.J) GO TO 25
246        28 CONTINUE
```

```
      29 IY(J)=IX(J)+ITETA
         GO TO 20
      25 IY(J)=IX(J)
      20 CONTINUE
         ALFA2=DIF(XLAG,IY,I)
         IF(ALFA2) 11,12,12
      11 ICOUN=ICOUN+1
         ALFA1=DIF(XLAG,IX,I)
         RATIO=ALFA2/ALFA1
         ISIR(ICOUN)=I
         IF(ICOUN.EQ.1) GO TO 41
         IF(RATIO.LT.RMAX) GO TO 10
      41 RMAX=RATIO
         JJ=I
         GO TO 10
      12 NOZE=NOZE+1
         JARR(NOZE)=I
      10 CONTINUE
         IF(IKODE.EQ.1) GO TO 110
         IF(ICOUN.EQ.0) GO TO 110
         DO 30 I=1,N
         IF(I.EQ.JJ) GO TO 33
         GO TO 35
      33 IF(ITETA.EQ.-1.AND.IX(I).EQ.0) GO TO 35
         IY(I)=IX(I)+ITETA
         GO TO 30
      35 IY(I)=IX(I)
      30 CONTINUE
         IF(ICOUN.EQ.1) GO TO 60
         NUC=0
         DO 50 J=1,ICOUN
         IF(ISIR(J).EQ.JJ) GO TO 50
         JL=ISIR(J)
C
C    PROVE BY CONTRADICTION THAT JL-COMPONENT CAN'T BE 1
         CALL PART1(IY,JJ,JL,IZ,ICODE)
         IF(ICODE.EQ.0) GO TO 55
         DO 59 J1=1,N
      59 IY(J1)=IZ(J1)
         GO TO 50
      55 NUC=NUC+1
         MOUC(NUC)=J
      50 CONTINUE
         IF(NUC.EQ.0) GO TO 60
         IF(NUC.GT.1) GO TO 40
         JUC=MOUC(NUC)
         IF(ITETA.EQ.-1.AND.IY(JUC).EQ.0) GO TO 60
         IY(JUC)=IY(JUC)+ITETA
         GO TO 60
      40 CALL PART2(NUC,MOUC,IY,XLB)
         IF(XLB.GE.VX) GO TO 100
         CALL PART3(IY,VX,JJ,NUC,MOUC,IZ)
         DO 61 J1=1,N
      61 IY(J1)=IZ(J1)
      60 VY=VNORS(XLAG,IY)
         IF(VY.GE.VX) GO TO 100
         VXO=VY
         DO 65 I=1,N
      65 IXO(I)=IY(I)
         GO TO 300
```

```
305       100 ICOUN=ICOUN-1
306           IF(ICOUN.EQ.0) GO TO 110
307           NOZE=NOZE+1
308           JARR(NOZE)=JJ
309           IKODE=1
310           GO TO 400
311       110 IF(ITETA.EQ.-1) GO TO 200
312           ITETA=-1
313           GO TO 500
314       200 DO 70 I=1,N
315        70 IXO(I)=IX(I)
316           VXO=VX
317       300 RETURN
318           END

319           SUBROUTINE PART1(IX,JJ,JL,IY,ICODE)
      C
      C   PERFORMS A TEST BY SIMPLY EVALUATING AN UPPER BOUND FOR THE LINEAR
      C    IF ICODE=1 , TEST SUCCEEDED
      C    IF ICODE=0 TEST FAILED
      C
320           DIMENSION IX(100),IY(100),IZ(100),XC(50),XA(50),JARR(100)
321           COMMON N,M/BLOCK5/ITETA/BLOCK6/WLAM/BLOCK8/NOZE,JARR
322           COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
             *NPA2(50),NPA3(50),NVNOR/TAB/POISON(5),100)
      C
323           NPA1(NITER)=NPA1(NITER)+1
324           M1=M+1
325           XLAG=WLAM
326           B0=-DIF(XLAG,IX,JL)
327           IF(B0.GE.0.0) GO TO 50
328           DO 10 I=1,N
329           IF(I.EQ.JJ) GO TO 11
330           IZ(I)=IX(I)
331           GO TO 10
332        11 IZ(I) IX(I)-ITETA
333        10 CONTINUE
334           BI=-DIF(XLAG,IZ,JJ)
335           DO 20 I=1,M1
336           IXS=IX(JL)+I
337           T1=PDFOP(IXS,JL)
338           T2=POISON(IXS,JL)
339           XC(I)=-T1/T2
340           IXP=IX(JJ)+I+1
341           P1=PDFOP(IXP,JJ)
342           P2=POISON(IXP,JJ)
343           XA(I)=-P1/P2
344        20 CONTINUE
345           TER1=1.
346           TER2=1.
347           IF(BI.LT.0.0) GO TO 22
348           W1=0.0
349           GO TO 23
350        22 W1=(1.1)*B0/BI
351        23 TERM=W1*BI
352           DO 30 I=1,M1
353           WI=XC(I)-W1*XA(I)
354           IF(WI) 30,30,35
355        35 DO 40 K=1,N
356           DO 37 KK=1,NOZE
```

```
      IF(JARR(KK).EQ.K) GO TO 36
   37 CONTINUE
      IXK=IX(K)+I+1
      TER1=TER1*POISON(IXK,K)
      GO TO 38
   36 IXK=IX(K)+I
      TER1=TER1*POISON(IXK,K)
   38 IF(K.EQ.JJ) GO TO 39
      IYK=IX(K)+I-1
      GO TO 41
   39 IYK=IX(K)+I+1
   41 TER2=TER2*POISON(IYK,K)
   40 CONTINUE
      UI=TER1-TER2
      TERM=TERM+UI*WI
   30 CONTINUE
      WW=TERM-80
      ICODE=0
      IF(WW) 50,55,55
   50 ICODE=1
      IF(ITETA.EQ.-1.AND.IX(JL).EQ.0) GO TO 55
      IX(JL)=IX(JL)+ITETA
   55 DO 60 I=1,N
   60 IY(I)=IX(I)
      RETURN
      END

      SUBROUTINE PART2(NUC,MOUC,IY,XLB)
C
C    IF PART1 FAILS, THIS FINDS AN UPPER BOUND FOR THE SUM OF FIRST
C    DIFFERENCES OF THE UNDETERMINED COEFFICIENTS
C
      DIMENSION IY(100),MOUC(100),IZ(100)
      COMMON N,M/BLOCK5/ITETA/BLOCK6/WLAM
      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
     *NPA2(50),NPA3(50),NVNOR
C
      NPA2(NITER)=NPA2(NITER)+1
      SUM=0.0
      DO 10 I=1,NUC
      IXK=MOUC(I)
      IXK1=IXK-1
      IF(IXK1.EQ.0) GO TO 15
      DO 20 J=1,IXK1
      IF(ITETA.EQ.-1.AND.IY(J).EQ.0) GO TO 25
      IZ(J)=IY(J)+ITETA
      GO TO 20
   25 IZ(J)=IY(J)
   20 CONTINUE
      NXK=IXK
      GO TO 35
   15 NXK=1
   35 DO 30 J=NXK,N
   30 IZ(J)=IY(J)
      TERM=DIF(WLAM,IZ,IXK)
      IF(TERM.LT.0.0) GO TO 40
      TERM=0.0
   40 SUM=SUM+TERM
   10 CONTINUE
      XLB=SUM+VNORS(WLAM,IY)
```

```
      RETURN
      END

      SUBROUTINE PART3(IX,VX,JJ,NUC,MOUC,IF)
C
C   PERFORMS THE SAME TEST AS PART1, BUT SETS UP THE LINEAR PROGRAM EX
C
      DIMENSION IX(100),IY(100),IZ(100),JARR(100),MOUC(100),KOUC(100),
     *10,10)
      COMMON N,M/BLOCK5/ITETA/BLOCK8/NOZE,JARR/BLOCK9/NROW,NCOL/BLOCK6
     *,LAM
      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
     *,NPA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
      NPA3(NITER)=NPA3(NITER)+1
C
      NUCB=NUC
      XLAG=WLAM
      NROW=M+2
      NCOL=M+3
      NCOL1=NCOL-1
      DO 9 I=1,N
      IF(I.EQ.JJ) GO TO 8
      IZ(I)=IX(I)
      GO TO 9
    8 IZ(I)=IX(I)-ITETA
    9 CONTINUE
      DO 10 ISAY=1,NUCB
      JL=MOUC(ISAY)
      Y(1,1)=0.0
      DO 11 J=2,NCOL1
      TER1=1.0
      TER2=1.0
      DO 40 K=1,N
      DO 37 KK=1,NOZE
      IF(JARR(KK).EQ.K) GO TO 36
   37 CONTINUE
      IXK=IZ(K)+J
      GO TO 38
   36 IXK=IZ(K)+J-1
   38 TER1=TER1*POISON(IXK,K)
      IF(K.EQ.JJ) GO TO 39
      IYK=IZ(K)+J-1
      GO TO 41
   39 IYK=IZ(K)+J
   41 TER2=TER2*POISON(IYK,K)
   40 CONTINUE
   11 Y(1,J)=TER1-TER2
      Y(1,NCOL)=-DIF(XLAG,IZ,JJ)
      DO 15 I=2,NROW
      IXS=IX(JL)+I-1
      Y(I,1)=PDFOP(IXS,JL)/POISON(IXS,JL)
      IXP=IX(JJ)+I-2
      IXR=IX(JJ)+I-1
   15 Y(I,NCOL)=PDFOP(IXP,JJ)/POISON(IXP,JJ)
      DO 25 I=2,NROW
      DO 25 J=2,NCOL1
      IF(J.EQ.I) GO TO 27
      Y(I,J)=0.0
      GO TO 25
   27 Y(I,J)=-1.0
```

```
462       25 CONTINUE
463          BO=-DIF(XLAG,IX,JL)
464          DO 210 I=1,NROW
465         210 CONTINUE
466          CALL XLPRO(Y,VOBJ)
467          FARK=VOBJ-BO
468          IF(FARK.GT.0.0) GO TO 10
469          IF(IX(JL).EQ.0.AND.ITETA.EQ.-1) GO TO 50
470          IX(JL)=IX(JL)+ITETA
471       50 KUC=NUC-1
472          IF(KUC.EQ.0) GO TO 10
473          NKUC=0
474          DO 55 J=1,NUC
475          IF(MOUC(J).EQ.JL) GO TO 56
476          IF(NKUC.EQ.1) GO TO 57
477          KOUC(J)=MOUC(J)
478          GO TO 55
479       56 NKUC=1
480          GO TO 55
481       57 J1=J-1
482          KOUC(J1)=MOUC(J)
483       55 CONTINUE
484          CALL PART2(KUC,KOUC,IX,XLB)
485          IF(XLB.GE.VX) GO TO 500
486          NUC=KUC
487          DO 60 I=1,NUC
488       60 MOUC(I)=KOUC(I)
489       10 CONTINUE
490      500 DO 70 I=1,N
491       70 IY(I)=IX(I)
492          RETURN
493          END

494          FUNCTION DIF(XLAG,IX,K)
         C
         C    CALCULATES THE FIRST DIFERENCES
         C
495          DIMENSION IX(100)
496          COMMON N,M/BLOCK1/C(100)/TAB/POISON(50,100)
497          DOUBLE PRECISION PRO,SUM
498          M1=M+1
499          SUM=0.D+00
500          DO 20 J1=1,M1
501          PRO=1.D+00
502          DO 30 I=1,N
503          IXJ=IX(I)+J1
504          IF(I.EQ.K) GO TO 25
505          PRO=PRO*POISON(IXJ,I)
506          GO TO 30
507       25 PRO=PRO*PDFOP(IXJ,I)
508       30 CONTINUE
509       20 SUM=SUM+PRO
510          IXM=IX(K)+M+2
511          DIF=-SUM-1.+POISON(IXM,K)+XLAG*C(K)
512          RETURN
513          END

514          FUNCTION VNORS(XLAG,IX)
         C
         C    EVALUATES THE FUNCTION E(NORS/X)+L.C.X
```

```
      C
515         DIMENSION IX(100),PM(100,20),PRO(20)
516         COMMON N,M
517         COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
           *NPA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
518         DOUBLE PRECISION SUM,PRO,TOPL,PM,XW
519         NVNOR=NVNOR+1
520         EPS=.00001
521         M1=M+1
522         SUM=0.D+00
523         DO 20 JJ=1,M1
524         PRO(JJ)=1.D+00
525         DO 25 II=1,N
526         IXJ=IX(II)+JJ-1
527         IF(IXJ.GT.0) GO TO 24
528         XX=PDFOP(IXJ,II)
529         GO TO 25
530      24 XX=POISON(IXJ,II)
531      25 PRO(JJ)=PRO(JJ)*XX
532      20 SUM=SUM+1.-PRO(JJ)
533         TOPL=0.D+00
534         DO 30 I=1,N
535         K=0
536      40 MXK=IX(I)+M+2+K
537         WX=PDFOP(MXK,I)
538         WW=FLOAT(K+1)*WX
539         IF(WX.LT.EPS) GO TO 30
540         TOPL=TOPL+WW
541         K=K+1
542         GO TO 40
543      30 CONTINUE
544         VNORS=SUM+TOPL+XLAG*DOTPR(IX)
545         RETURN
546         END

547         SUBROUTINE TABLE(POISON)
      C
      C   CALCULATES THE CUMULATIVE POISSON SUMS
      C
548         DIMENSION POISON(50,100)
549         COMMON N,M/BLOCK2/XLAM(100)
550         DOUBLE PRECISION EPS,TERM,SUMX
551         EPS=1.D-06
552         DO 10 J=1,N
553         TERM=1.D+00
554         SUMX=1.D+00
555         DO 20 I=1,50
556         TERM=TERM*XLAM(J)/FLOAT(I)
557         IX=I
558         IF(TERM.LT.EPS) GO TO 25
559         SUMX=SUMX+TERM
560         POISON(I,J)=EXP(-XLAM(J))*SUMX
561         IF(POISON(I,J).GT.1.0) POISON(I,J)=1.0
562      20 CONTINUE
563      25 IF(IX.EQ.50) GO TO 10
564         DO 30 I=IX,50
565      30 POISON(I,J)=1.0
566      10 CONTINUE
567         RETURN
568         END
```

```
      FUNCTION PDFOP(K,I)
C
C  CALCULATES THE INDIVIDUAL POISSON TERMS
C
      COMMON /BLOCK2/XLAM(100)
      DOUBLE PRECISION SUM,WW
      SUM=0.D+00
      IF(K.LT.2) GO TO 10
      DO 15 J=2,K
      SUM=SUM+ALOG(FLOAT(J))
   15 CONTINUE
   10 WW=-XLAM(I)+K*ALOG(XLAM(I))-SUM
      IF(WW.LT.-15.0) GO TO 20
      PDFOP=DEXP(WW)
      GO TO 25
   20 PDFOP=0.0
   25 RETURN
      END

      FUNCTION DOTPR(IX)
C
C  CALCULATES THE DOT PRODUCT C.X
C
      DIMENSION IX(100)
      COMMON N,M/BLOCK1/C(100)
      SUM=0.0
      DO 10 I=1,N
      XX=FLOAT(IX(I))
   10 SUM=SUM+XX*C(I)
      DOTPR=SUM
      RETURN
      END

      SUBROUTINE SIRALA(X,IR)
C
C  THIS ARRANGES AN ARRAY OF ELEMENTS IN ASCENDING ORDER
C  IR(K) IS THE RANK OF ITEM K INTHE LIST
C
      DIMENSION X(100),IR(100),ISEQ(100),Y(100)
      COMMON N,M
      DO 9 I=1,N
    9 ISEQ(I)=I
      DO 10 I=1,N
      XMIN=X(I)
      IMINJ=I
      IMIN=ISEQ(I)
      N1=N-I+1
      DO 20 J=1,N1
      IF(X(J).LT.XMIN) GO TO 21
      GO TO 20
   21 XMIN=X(J)
      IMIN=ISEQ(J)
      IMINJ=J
   20 CONTINUE
      IR(IMIN)=I
      IF(N1.EQ.1) GO TO 10
      N2=N1-1
      DO 30 J=1,N2
      IF(J.LT.IMINJ) GO TO 33
      J1=J+1
```

```
      ISEQ(J)=ISEQ(J1)
      Y(J)=X(J1)
      GO TO 29
   33 Y(J)=X(J)
   29 X(J)=Y(J)
   30 CONTINUE
   10 CONTINUE
      RETURN
      END

      SUBROUTINE XLPRO(Y,VOBJ)
C
C   SOLVES A LINEAR PROGRAM OF THE TYPE
C   MAX CX ST AX<B,X>0
C
      DIMENSION Y(10,20),X(10,20),ISNB(10),ISBV(10),MROW(10)
      COMMON /BLOCK9/NROW,NCOL
C
      ISBV(1)=0
      ISNB(1)=NCOL+1
      DO 10 I=2,NROW
   10 ISBV(I)=I-2+NCOL
      DO 11 J=2,NCOL
   11 ISNB(J)=J-1
      NROW1=NROW-1
      NCOL1=NCOL-1
      NCOL2=NCOL+1
      NCOL3=NCOL+NROW1
      DO 12 I=1,NROW
      DO 12 J=NCOL2,NCOL3
      IM=I-1
      JM=J-NCOL
      IF(JM.EQ.IM) GO TO 13
      Y(I,J)=0.0
      GO TO 12
   13 Y(I,J)=1.0
   12 CONTINUE
  100 NNC=0
      NPV=0
      NPC=0
      NNV=0
      DO 15 J=2,NCOL
      IF(Y(1,J).LT.0.0) GO TO 16
      NNC=NNC+1
      IF(NNC.EQ.NCOL1) GO TO 500
      GO TO 15
   16 NPC=NPC+1
      IF(NPC.EQ.1) GO TO 17
      IF(Y(1,J).GT.YMIN) GO TO 15
   17 YMIN=Y(1,J)
      JMIN=J
   15 CONTINUE
      IEV=ISNB(JMIN)
      NRVA=1
      DO 20 I=2,NROW
      IF(Y(I,JMIN).LE.0.0) GO TO 25
      NNV=NNV+1
      TETA=Y(I,1)/Y(I,JMIN)
      IF(NNV.EQ.1) GO TO 23
      IF(TETA-TMIN) 23,20,20
```

```
671        23 TMIN=TETA
672           IMIN=I
673           MROW(NMVA)=IMIN
674           GO TO 20
675        25 NPV=NPV+1
676           IF(NPV.EQ.NROW1) GO TO 450
677        20 CONTINUE
678           DO 26 I=2,NROW
679           IF(Y(I,JMIN).LE.0.0) GO TO 26
680           TETA=Y(I,1)/Y(I,JMIN)
681           IF(TETA.GT.TMIN) GO TO 26
682           IF(I.EQ.IMIN) GO TO 26
683           NMVA=NMVA+1
684           MROW(NMVA)=I
685        26 CONTINUE
686           IF(NMVA.EQ.1) GO TO 200
687           DO 80 I=1,NMVA
688           IK=MROW(I)
689           JPOS=0
690           DO 80 J=NCOL2,NCOL3
691           X(IK,J)=Y(IK,J)/Y(IK,JMIN)
692           IF(X(IK,J).GT.0.0) GO TO 85
693           GO TO 80
694        85 JPOS=JPOS+1
695           IF(JPOS.GT.1) GO TO 80
696           IF(IK.GT.MROW(1)) GO TO 87
697           GO TO 93
698        87 IF(J-JKMIN) 80,90,93
699        90 IF(X(IMIN,JKMIN).LE.X(IK,J)) GO TO 80
700        93 IMIN=IK
701           JKMIN=J
702        80 CONTINUE
703       200 IDV=ISBV(IMIN)
704           ISNB(JMIN)=IDV
705           ISBV(IMIN)=IEV
706           DO 40 I=1,NROW
707           DO 40 J=1,NCOL3
708           IF(I.EQ.IMIN) GO TO 55
709           IF(J.EQ.JMIN) GO TO 60
710           X(I,J)=Y(I,J)-(Y(I,JMIN)*Y(IMIN,J)/Y(IMIN,JMIN))
711           GO TO 40
712        55 IF(J.EQ.JMIN) GO TO 57
713           X(I,J)=Y(I,J)/Y(IMIN,JMIN)
714           GO TO 40
715        57 X(I,J)=1./Y(I,J)
716           GO TO 40
717        60 X(I,J)=-Y(I,J)/Y(IMIN,JMIN)
718        40 CONTINUE
719           DO 70 I=1,NROW
720           DO 70 J=1,NCOL3
721        70 Y(I,J)=X(I,J)
722           GO TO 100
723       450 WRITE(6,250)
724       250 FORMAT(5X,'UNBOUNDED SOLUTION')
725       500 RETURN
726           END

      $ENTRY
```

```
THE SADDLE POINT SEARCH ALGORITHM


PROBLEM NUMBER   1

NUMBER OF ITEMS=    5  BUDGET=$  25000.00


ITEM    COST/UNIT        DEMAND      INITIAL SOLUTION     OPTIMAL SOLUTION

#  1      2980.00          2.10              2                   2
#  2      1751.00          1.50              2                   2
#  3       462.00          1.20              2                   3
#  4      1500.00          5.00              6                   7
#  5       345.00          3.50              6                   6

E(NORS/X)=    1.06282     COST(X)=$  23418.00

LAGRANGE MULTIPLIER=  0.000090

# OF ITERATIONS=   6

# OF NORS EVALUATIONS= 141

# OF  STEPS IN EACH ITERATION

ITERATION #     STEP1  STEP2  STEP3  STEP4  PART1  PART2  PART3
          1         1      1      1      1      5      0      0
          2         1      5      5      2      0      0      0
          3         1      4      4      2      3      0      0
          4         1      2      2      1      1      0      0
          5         1      2      2      1      0      0      0
          6         1      2      2      1      2      0      0

STATEMENTS EXECUTED=  273427

CORE USAGE       OBJECT CODE=   31504 BYTES, ARRAY AREA=   40572 BYTES, TOTAL

DIAGNOSTICS        NUMBER OF ERRORS=       0, NUMBER OF WARNINGS=       0, N

COMPILE TIME=     0.52 SEC, EXECUTION TIME=     5.02 SEC,     12.27.33     TUE

      C$STOP
```

APPENDIX B

B.1 Detail Documentation of Greedy Algorithm Computer Program

B.2 FORTRAN Listing of Greedy Algorithm Computer Program and Solution Output

APPENDIX B.1

DETAIL DOCUMENTATION OF

GREEDY ALGORITHM

COMPUTER PROGRAM

APPENDIX B.1

DETAIL DOCUMENTATION OF GREEDY ALGORITHM COMPUTER PROGRAM

General

In addition to the explanatory comment statements that are provided in the program listing of the algorithm (See Appendix B.2), formulations of computational formulas, and detailed statement by statement interpretations are provided in this appendix for several critical parts of the program. Since some of the subroutines are either essentially the same as those of the D029 program, or rather simple and straight forward, the effort below is devoted mainly to three subroutines: TABLE, OPTIM, and MARGAN. A flow chart describing the relationship between OPTIM and MARGAN is also provided.

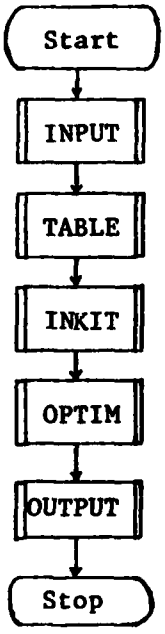
Terminology and Notations Used in Notes on Subprograms

a_i = NAP(I)	number of applications of item i
b_i = BO(I+1)	conditional operational rate, given i planes available for canibilization
q_i = KIT(I)	quantity of item i in the kit
w	weighting factor of $\Delta E(SDO)$ used in r_i calculation
$F_i(x)$	probability of demand $\leq$ x for item i
ITOT	total number of items considered
MUP	upper bound of subscript of b_i
NAP(I)=a_i	
$NMAX_i$	maximum possible number of demands for item i
NOPS = UE	number of planes in squadran
NORS = NMCS	number of Not Operationally Ready Supply aircraft
POISON (I,J)	cumulative Poisson demand $F_i(j-1)$ for item i

r_i=RATIO(I)	Weighted average improvement of E(NORS) & E(SDO) per dollar invested
SDO	number of Stock Due Outs, or backorders
S(N)	probability of NORS $\leq$ N-1
UE = NOPS	

Notes on Main Program

This main program contains the various subroutine calling statements shown below.



Notes on Subroutine INPUT

This subroutine reads input data, calculates average demand rates, and obtains KIT 5718 and its target value. The code is essentially the same as the code for the DO29 algorithm. LRU-RR, LRU-RRR, SRU-RR, and SRU-RRR are differentiated and processed in different manners as required.

Notes on Subroutine TABLE

This subroutine calculates the individual cumulative Poisson probabilities for each item.

Poisson probability is $P_I(x) = \frac{\lambda_I^x e^{-\lambda_I}}{x!}$, x = 0,1,2,... and I = 1,2,...ITOT

hence, $P_I(x+1) = \frac{\lambda_I}{x+1} P_I(x)$

e.g. $P_I(0) = e^{-\lambda_I}$, $P_I(1) = \lambda_I \cdot P_I(0)$, $P_I(2) = \frac{\lambda}{2} P_I(1)$,...etc.

Then the cumulative Poisson probability, $F_I(x)$, can be expressed as follows:

$$F_I(x) = \sum_{j=0}^{x} \frac{\lambda_I^j e^{-\lambda_I}}{j!} = F_I(x-1) + P_I(x)$$

By employing the two dimensional array POISON (500,60), $F_i(j)$ of item i can be stored in POISON (I, j+1), since POISON (I, 1) = $F_I(0)$ for item I.

The main segment of code for this subroutine is shown below.

```
ISN 0007            TERL=5.D-12
ISN 0008            DO 10 I=1,ITOT
ISN 0009            WWW=DBLE(XMU(I))
ISN 0010            TERM=DEXP(-WWW)
ISN 0011            POISON(I,1)=TERM
ISN 0012            DO 20 J=2,60
ISN 0013            IF(TERM.LT.TERL.AND.POISON(I,J-1).GT.9.D-01) GO TO 30
ISN 0015            TERM=TERM*WWW/DFLOAT(J-1)
ISN 0016            POISON(I,J)=POISON(I,J-1)+TERM
ISN 0017         20 CONTINUE
ISN 0018            GO TO 10
ISN 0019         30 DO 40 JX=J,60
ISN 0020         40 POISON(I,JX)=1.D+00
ISN 0021         10 CONTINUE
ISN 0022            RETURN
ISN 0023            END
```

Line 7 gives the criteria for stopping further computations of the upper-tail of the Poisson distribution. At lines 10 and 11, POISON (I, 1) = $F_I(0)$ = $e^{-\lambda_I}$. Line 13 checks if the stopping criteria is met. Lines 15 and 16 calculate

all $P_I(x)$ and $F_I(x)$ for each item. Lines 19 and 20 assign to all the remaining cumulative probability terms the value of 1.

Notes on Subroutine INKIT

This subroutine is essentially the same as that of D029. It appropriately increases KIT quantities to ensure that the probability of any item being the cause of half a squadron being down is less than 0.50.

Notes on Subroutine OPTIM

This routine calls marginal analysis subroutine, MARGAN, and updates the parameters b_i's. It terminates if the new set of b_i's are sufficiently close to the values of the previous set.

$$BO(J) = b_{j-1} = \prod_{\ell=1}^{ITOT} F_\ell \left[q_\ell * + a_\ell (j-1)\right]$$

$$= \prod_{\ell=1}^{ITOT} \text{POISON} \left[KIT(\ell) + NAP(\ell) * (J-1) + 1\right]$$

The measure of closeness between two sets of b_i's and b_i''s is the sum of squares $= \sum_i (b_i - b_i')^2$.

Referring to the code of the main body of this subroutine on the next page and the combined flow chart that follows, the algorithm can be clearly explained.

```
                C
ISN 0017          100 CALL MARGAN(KIT,PRICE)
                C
ISN 0018              NITER=NITER+1
                C
                C        STEP 2
                C        UPDATE THE PARAMETERS B
                C
ISN 0019              SUMSQ=0.D+00
ISN 0020              DO 10 J=1,MUP
ISN 0021              PRO=1.D+00
ISN 0022              DO 20 I=1,ITOT
ISN 0023              KITPJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0024              IF(KITPJ.GT.60) GO TO 20
ISN 0026              PRO=PRO*POISON(I,KITPJ)
ISN 0027              IF(PRO.LT.EPSQ) GO TO 15
ISN 0029           20 CONTINUE
ISN 0030           15 SUMSQ=SUMSQ+((BO(J)-PRO)**2)
ISN 0031              BO(J)=PRO
ISN 0032           10 CONTINUE
                C
                C        STEP 3
                C        COMPARE THE NEW VALUES OF B WITH PREVIOUS VALUES
                C        TERMINATE IF THEY ARE SUFFICIENTLY CLOSE
                C
ISN 0033              IF(SUMSQ.LT.EPS) GO TO 30
ISN 0035              GO TO 100
                C
ISN 0036           30 CALL CALC(KIT,ENDRS,SCO)
ISN 0037              RETURN
ISN 0038              END
```

Line 17 calls the marginal analysis subroutine in order to find the optimum kit under the current b_i parameters. Once this is done, it comes back and starts to update the b_i's.

Line 24 provides a safeguard against running out of the limit of the Poisson array. At line 26, b_i's are calculated by multiplying all pertinent cumulative probability terms together. Line 27 gives a lower bound check on b_i to save further calculations, since when $b_i \leq \text{EPSQ} = 5\text{x}10^{-11}$, we can treat it as zero without practically affecting actual results. Line 30 gives a measure of the difference between the latest two consecutive sets of b_i's. Line 33 checks to see if the difference is small enough to stop. If not, the

algorithm continues iterations until the above stopping criteria is satisfied. Finally, before returning to the main program, the performance of the optimal kit is evaluated.

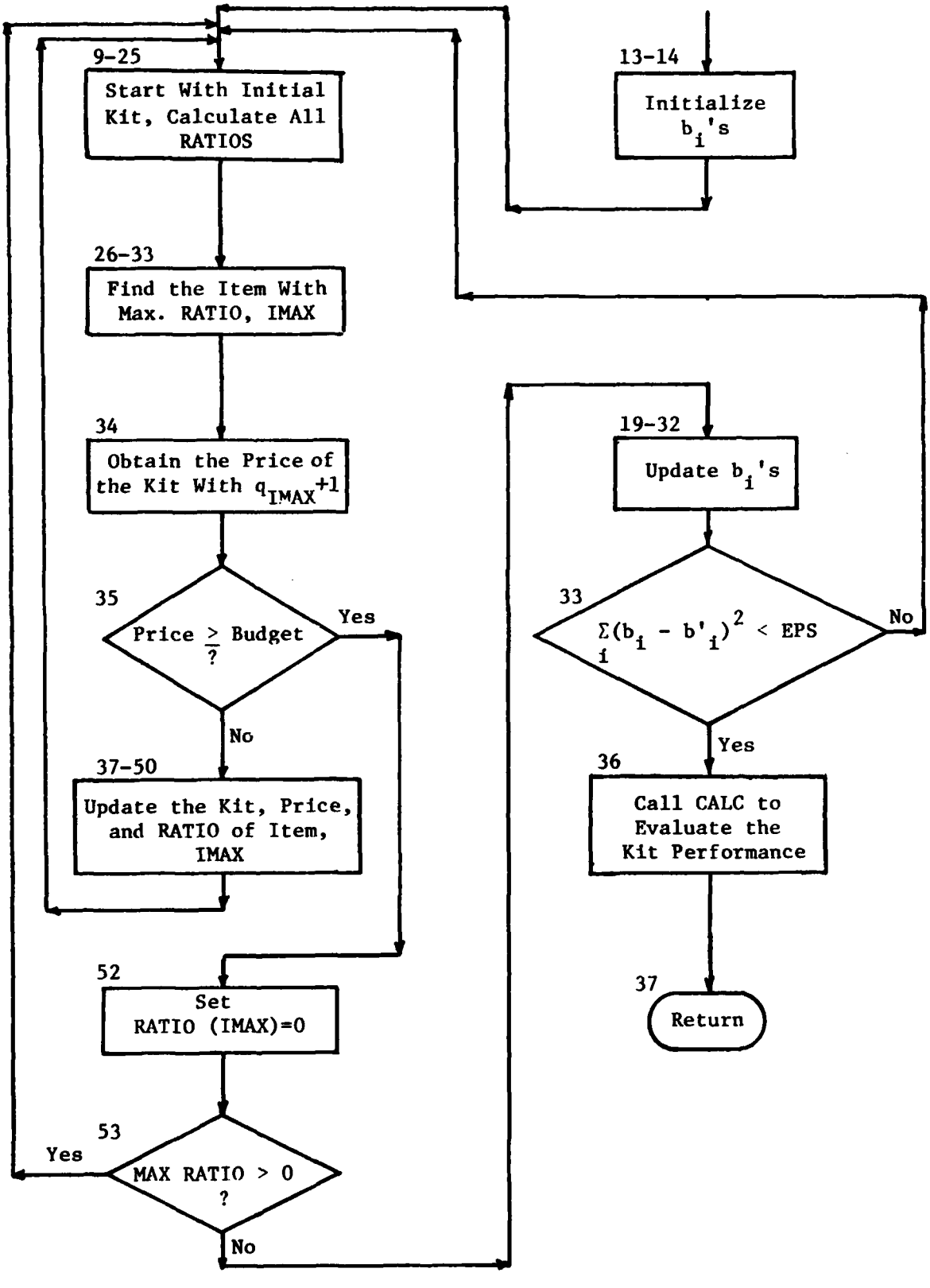


Figure B.1. Flowcharts for Subroutines MARGAN and OPTIM

Notes on Subroutine MARGAN

This subroutine performs the marginal analysis. It starts with the initial kit and calculates the RATIO of return, (r_i), per dollar invested for each item i. This ratio is the weighted average of the improvements of E(NORS) and E(SDO).

$$r_i = \frac{\Delta_i E(NORS) + w\Delta_i E(SDO)}{c_i}$$

$$\text{where } \Delta_i E(NORS) = \left[\sum_{\ell=1}^{ITOT} (MUP_\ell + 1) - \sum_{j=0}^{MUP} b_j \sum_{\ell=1}^{ITOT} \ell n\ F_\ell(q_\ell + ja_\ell)\right]$$

$$- \left[\sum_{\ell=1}^{ITOT} (MUP_\ell + 1) - \sum_{j=0}^{MUP} b_j \sum_{\ell=1}^{ITOT} \ell n\ F_\ell(q'_\ell + ja_\ell)\right]$$

$$\text{where } q'_\ell = q_\ell + 1 \text{ when } \ell = i$$

$$q'_\ell = q_\ell \qquad \text{when } \ell \neq i$$

$$\Delta_i E(NORS) = \sum_{j=0}^{MUP} b_j \left[\ell n\ F_i\ (q_i + 1 + ja_i) - \ell n\ F_i\ (q_i + ja_i)\right]$$

$$= \sum_{j=0}^{MUP} b_j\ \ell n \frac{F_i(q_i + ja_i + 1)}{F_i(q_i + ja_i)}$$

$$= \sum_{j=0}^{MUP} b_j\ \ell n \frac{POISON\ (i,\ q_i + ja_i + 2)}{POISON\ (i,\ q_i + ja_i + 1)}$$

Since POISON (I, 1) = F_i (0)

$$E(SDO) = \sum_{\ell=1}^{ITOT} \sum_{x=q_\ell}^{NMAX_\ell - 1} [1-F_\ell(x)] = \sum_{\ell=1}^{ITOT} \sum_{j=0}^{JUP_\ell} [1-F_\ell(q_\ell + j)]$$

where JUP_ℓ = (NMAX - 1) - q_ℓ

= [KIT (ℓ) + (NOPS-1)*NAP(ℓ) + 1] - 1 - KIT(ℓ)

= (NOPS-1)*NAP(ℓ) + 1

$$\text{Then } \Delta_i E(SDO) = \sum_{\ell=1}^{ITOT} \sum_{j=0}^{JUP_\ell} [1-F_\ell(q_\ell+j)] - \sum_{\ell'=1}^{ITOT} \sum_{j=0}^{JUP} [1-F_{\ell'}(q_{\ell'}+j)]$$

where $q_{\ell'} = q_\ell + 1$ when $\ell = i$

$q_{\ell'} = q_\ell$ when $\ell \neq i$

$$= \sum_{j=0}^{JUP_i} \left\{1-F_i(q_i + j) - [1-F_i(q_i + 1 + j)]\right\}$$

$$\text{i.e. } \Delta_i E(SDO) = \sum_{j=0}^{JUP_i} [F_i(q_i + 1 + j) - F_i(q_i + j)]$$

$$= F_i(q_i + 1 + JUP_i) - F_i(q_i)$$

$$\text{or} = POISON\ (i, q_i + 1 + NMAX_i - 1 - q_i + 1) - POISON\ (i, q_i + 1)$$

$$= POISON\ (i, NMAX_i + 1) - POISON\ (i, KIT(i) + 1)$$

After comparing each ratio r_i and obtaining the maximum one, we increase its kit quantity by one. Termination occurs if the kit's price is greater than budget, or the procedure is repeated to update the kit's value and the ratio until the budget is exceeded.

Ratio Calculation Segment:

```
C          START WITH THE INITIAL KIT AND CALCULATE THE RATIO
C          PER DOLLAR INVESTED FOR EACH ITEM
C
ISN 0009        PRICE=PRCIN
ISN 0010        DO 10 I=1,ITOT
ISN 0011        KIT(I)=KITIN(I)
ISN 0012        NMAX1=KIT(I)+(NOPS-1)*NAP(I)+2
ISN 0013        IF(NMAX1.GE.60) NMAX1=60
ISN 0015        IXS=KIT(I)+1
ISN 0016        SUM=DBLE(PAR)*(POISON(I,NMAX1)-POISON(I,IXS))
ISN 0017        DO 20 J=1,MUP
ISN 0018        IXJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0019        IF(IXJ.GE.60) GO TO 21
ISN 0021        WW=POISON(I,IXJ+1)/POISON(I,IXJ)
ISN 0022        GO TO 20
ISN 0023     21 WW=1.D+00
ISN 0024     20 SUM=SUM+BO(J)*DLOG(WW)
ISN 0025     10 RATIO(I)=SUM/DBLE(COST(I))
```

Line 9 and 11 start with the initial kit and its price. Line 15 increases KIT quantity by 1, once for every item. Line 16 gives the $w \cdot \Delta E(SDO)$, which is

the weighted improvement of E(SDO). Line 19 provides a safeguard against running out of the POISON array limits. Line 21 and 24 calculate $\Delta E(NORS)$, improvements of E(NORS), and add it to $W \cdot \Delta E(SDO)$. Line 25 computes the performance improvement measurement, RATIO.

The Remaining Segments:

```
                C
                C          FIND THE MAXIMUM RATIO
                C
ISN 0026          100 DO 25 I=1,ITOT
ISN 0027              IF( I.EQ.1) GO TO 30
ISN 0029              IF( RATIO(I).LT.RMAX) GO TO 25
ISN 0031           30 IMAX=I
ISN 0032              RMAX=RATIO(I)
ISN 0033           25 CONTINUE
ISN 0034              XPRICE=PRICE+COST(IMAX)
                C
                C          TERMINATE IF THE TARGETS ARE ACHIEVED
                C
ISN 0035              IF( XPRICE.GT.BUDGET) GO TO 50
                C
                C          IF NOT, UPDATE THE KIT VALUE AND THE RATIO
                C
ISN 0037              KIT(IMAX)=KIT(IMAX)+1
ISN 0038              PRICE=XPRICE
ISN 0039              NMAX1=KIT(IMAX)+(NOPS-1)*NAP(IMAX)+2
ISN 0040              IF( NMAX1.GT.60) NMAX1=60
ISN 0042              IXS=KIT(IMAX)+1
ISN 0043              SUM=DBLE(PAR)*(POISON(IMAX,NMAX1)-POISON(IMAX,IXS))
ISN 0044              DO 40 J=1,MUP
ISN 0045              IXJ=KIT(IMAX)+(J-1)*NAP(IMAX)+1
ISN 0046              IF( IXJ.GE.60) GO TO 45
ISN 0048              WW=POISON(IMAX,IXJ+1)/POISON(IMAX,IXJ)
ISN 0049           40 SUM=SUM+BO(J)*DLOG(WW)
ISN 0050           45 RATIO(IMAX)=SUM/DBLE(COST(IMAX))
ISN 0051              GO TO 100
ISN 0052           50 RATIO(IMAX)=0.D+00
                C
ISN 0053              IF( RMAX.GT.0.D+00) GO TO 100
ISN 0055              RETURN
ISN 0056              END
```

Lines 26 to 33 find the maximum RATIO value. Line 34 then calculates the corresponding price. At line 35, the budget is compared with kit price. If the kit price is exceeded, continue to check the feasibility of

other possible kits. If not exceeded, the kit value and the RATIO are updated. Lines 37 to 50 are similar to previous segments, lines 9 to 25, except this time calculation of RATIO is for the most promising item only.

With the updated RATIO value, this item enters competition with all other items again. If the kit with $q_{imax} + 1$ is infeasible, then line 52 sets its ratio to zero in order to eliminate $q_{imax} + 1$ from further investigation. Line 53 then checks if any other possibilities for consideration are left out. If all possibilities are exhausted, then imax = 0; hence the program exits from MARGAN.

Notes on Subroutine CALC

This subroutine calculates the expected NORS and expected STOCK DUE OUT values.

$$E(NORS) = \sum_{Y=1}^{UE} YP(NORS = Y) = \sum_{Y=1}^{UE} Y\left[P(NORS \leq Y) - P(NORS \leq Y-1)\right]$$

$$= UE \cdot P(NORS \leq UE) - \sum_{j=0}^{UE-1} P(NORS \leq j)$$

$$= UE \cdot \prod_{i=1}^{ITOT} F_i(q_i + UE \cdot a_i) - \sum_{j=0}^{UE-1} \prod_{i=1}^{ITOT} F_i(q_i + ja_i)$$

$$= UE * S(UE + 1) - \sum_{j=1}^{UE} S(j)$$

where $S(N) = P(NORS \leq N - 1)$

$$= \prod_{i=1}^{ITOT} F_i(q_i + (N-1)*a_i)$$

$$\text{or} = \prod_{i=1}^{ITOT} POISON\ (i,\ q_i + (N-1) * a_i + 1)$$

$$\text{i.e. } E(NORS) = NOPS * S(NOPS + 1) - \sum_{j=1}^{NOPS} S(j)$$

where NOPS=UE=number of planes in the squadron.

$$E(SDO) = \sum_{\ell=1}^{ITOT} \sum_{x=q_\ell}^{NMAX_\ell - 1} [1-F_\ell(x)] = \sum_{\ell=1}^{ITOT} \sum_{j=0}^{JUP_\ell} [1-F_\ell(q_\ell + j)]$$

where $JUP_\ell = (NMAX_\ell - 1) - q_\ell$

$= [(q_\ell + (NOPS-1) * a_\ell + 1) - 1] - q_\ell$

$= (NOPS - 1) * a_\ell + 1$

$$E(SDO) = \sum_{\ell=1}^{ITOT} \sum_{j=0}^{JUP_\ell} [1-POISON\ (\ \ell,\ q_\ell + j)]$$

E(NORS) Computation Segment:

```
ISN 0009              EPS=5.D-10
                C
                C        E(NORS/K) COMPUTATION
                C
ISN 0010              SUM=0.D+00
ISN 0011              S(NOPS+1)=1.D0
ISN 0012              DO 10 K=1,NOPS
ISN 0013              S(K)=1.D+00
ISN 0014              DO 20 I=1,ITOT
ISN 0015              IXJ=KIT(I)+(K-1)*NAP(I)+1
ISN 0016              IF(IXJ.GT.600) GO TO 20
ISN 0018              S(K)=S(K)*POISON(I,IXJ)
ISN 0019              IF(S(K).LT.EPS) GO TO 15
ISN 0021           20 CONTINUE
ISN 0022              SUM=SUM+S(K)
ISN 0023              GO TO 10
ISN 0024           15 S(K)=0.D+00
ISN 0025           10 CONTINUE
ISN 0026              VNORS=DFLOAT(NOPS)*S(NOPS+1)-SUM
ISN 0027              ENORS=VNORS
```

Line 18 gives the probability of NORS $\leq$ N-1, S(N). Line 19 provides a check for very small S(N) values to save further computational effort. Line 24 just sets any small value ($\leq 5 \times 10^{-10}$) to zero for practical purposes. At line 26, E(NORS) is calculated by using the formula shown previously.

E(SDO) Computation Segment:

```
                C          E( SDO/ X)   COMPUTATION
                C
ISN 0028                   SDOP=0.D+00
ISN 0029                   DO 50 I=1,ITOT
ISN 0030                   JUP=(NOPS-1)*NAP(I)+1
ISN 0031                   DO 60 J=1,JUP
ISN 0032                   IXJ=KIT(I)+J
ISN 0033                   IF( IXJ.GT.60) GO TO 50
ISN 0035                60 SDOP=SDOP+1.D+00-PJ ISON(I,IXJ)
ISN 0036                50 CONTINUE
ISN 0037                   SDO=SDOP
ISN 003d                   RETURN
ISN 0039                   END
```

Line 35 gives the calculation of E(SDO), which is also formulated in the above section. At lines 28 and 35, SDOP is used instead of SDO to avoid alternating the current value stored in SDO. Line 37 then updates the SDO value.

Notes on Subroutine OTPUT

This subroutine prints out the final results of the optimization. It includes: the optimal kit, its price and performance indices E(NORS) and E(SDO); parameters set b_i's; weighting factor of RATIO computation; kit 57-18 and its relevant values. Number of iterations is also printed.

APPENDIX B.2

FORTRAN LISTING OF

GREEDY ALGORITHM

COMPUTER PROGRAM

AND SOLUTION OUTPUT

```
C
C  ***************************************************
C  *                                                 *
C  *                                                 *
C  *       GREEDY  ALGORITHM                         *
C  *                                                 *
C  *                                                 *
C  ***************************************************
C
ISN 0002        DIMENSION KIT(500)
C
ISN 0003        CALL INPUT1(KIT,PRICE)
ISN 0004        CALL TABLE
ISN 0005        CALL INKIT(KIT,PRICE)
ISN 0006        CALL OPTIM(KIT,PRICE,ENORS,SCO)
ISN 0007        CALL OTPUT1 (KIT,PRICE,ENORS,SOO)
C
ISN 0008        CALL INPUT2(KIT,PRICE)
ISN 0009        CALL TABLE
ISN 0010        CALL INKIT(KIT,PRICE)
ISN 0011        CALL OPTIM(KIT,PRICE,ENORS,SCO)
ISN 0012        CALL OTPUT2 (KIT,PRICE,ENORS,SOO)
ISN 0013        STOP
ISN 0014        END
```

```
ISN 0002        SUBROUTINE INPUT1(KIT,PRICE)
ISN 0003        DIMENSION KIT(500),DFT(30)
ISN 0004        COMMON POISON(500,60),BO(50)
ISN 0005        COMMON /BLOCK1/ITOT,NITER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006        COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007        COMMON /BLOCK4/ KTSTAT(500),GOAL,SGOAL,BUDKT
ISN 0008        COMMON /PHASE/ IPHASE
ISN 0009        COMMON /INPUT/NDAYS,DFP,NSUD
ISN 0010        DOUBLE PRECISION POISON,BO
ISN 0011        DATA XRU/'LRU'/,RRR/'RHR'/,DFT/7*1.15,.53,.22*.52/
C
ISN 0012        IPHASE=1
ISN 0013        NITER=0
ISN 0014        PRICE=0.0
ISN 0015        ITEM=0
C
ISN 0016        READ(5,109) PAR
C
ISN 0017        READ(5,101) SFPD,NDAYS, DFP,NOPS,NSUD,SKF
C
ISN 0018        MUP=NOPS+1
ISN 0019     10 ITEM=ITEM+1
C
ISN 002C     20 READ(1,102,END=50) CAP,NRC,NAP(ITEM),TDR,COST(ITEM),TYP
C
C    ONLY LRU ITEMS WILL BE PROCESSED
C
ISN 0021        IF(TYP.EQ.XRU) GO TO 30
ISN 0023        GO TO 20
ISN 0024     30 XMU(ITEM)=0.0
ISN 0025        LENT=NSUD
C
ISN 0026        IF(CAP.EQ.RRR) LENT=NSUD+NRC
C
C    DEMAND AND KIT  VALUES
C
ISN 0028        DO 40 IDAY=1,LENT
ISN 0029     40 XMU(ITEM)=XMU(ITEM)+FLOAT(NAP(ITEM))*TDR*DFT(IDAY)
ISN 0030        KIT(ITEM)=0
ISN 0031        GO TO 10
ISN 0032     50 ITOT=ITEM-1
```

```
C
C          COMPUTATION OF THE REQUIREMENT KIT 5718 & TARGET VALUES
C
ISN 0033        BUDGET=0.0
ISN 0034        DO 60 I=1,ITOT
ISN 0035        KT5718(I)=IFIX(XMU(I)*SKF+.5)
ISN 0036        IF(KT5718(I).LT.1) KT5718(I)=1
ISN 0038        BUDGET=BUDGET+COST(I)*FLOAT(KT5718(I))
ISN 0039     60 CONTINUE
C
ISN 0040        BUDKT=BUDGET
ISN 0041        BUDGET=.9*BUDGET
C
ISN 0042    101 FORMAT(F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0043    102 FORMAT(T19,A3,I2,1A,I2,T32,F5.4,T46,F8.2,A3)
ISN 0044    105 FORMAT(I4)
ISN 0045    109 FORMAT(F10.0)
ISN 0046    110 FORMAT(1008.0)
ISN 0047        RETURN
ISN 0048        END

ISN 0002        SUBROUTINE OTPUT1(KIT,PRICE,ENORS,SDO)
ISN 0003        DIMENSION KIT(500),SN(4)
ISN 0004        COMMON POISON(500,60),BO(5)
ISN 0005        COMMON /BLOCK1/ITOT,NITER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006        COMMON /BLOCK2/ MAP(500),COST(500),XMU(500)
ISN 0007        COMMON /BLOCK4/ KT5718(500),VGOAL,SGOAL,BUDKT
ISN 0008        COMMON /PHASE/ IPHASE
C
ISN 0009        DOUBLE PRECISION PJISJN,BO
C
ISN 0010        ITEM=0
C
ISN 0011        WRITE(6,200)
ISN 0012        WRITE(6,210) ITOT
C
C       PRINT OUT THE TARGET VALUES
C
ISN 0013        WRITE(6,205) (KT5718(I),I=1,ITOT)
ISN 0014        WRITE(6,220) BUDKT ,VGOAL,SGOAL
ISN 0015        WRITE(6,230) BUDGET.
C
ISN 0016        WRITE(6,100) PRICE,ENORS,SDO
ISN 0017        WRITE(6,110) NITER,(BO(J),J=1,MUP)
ISN 0018        WRITE(6,109) PAR
ISN 0019        WRITE(6,105)
ISN 0020        WRITE(6,120) (KIT(I),I=1,ITOT)
C
ISN 0021        RETURN 1
C
ISN 0022    100 FORMAT(5X,'PRICE(X)=$',F15.2,5X,'E(NORS/X)=',F10.5,5X,'E(SDO/X)=',
           *F10.5,///)
ISN 0023    105 FORMAT(///,5X,'INITIAL KIT',///)
ISN 0024    109 FORMAT(///,5X,'WEIGHT $',F10.3,///)
ISN 0025    110 FORMAT(/,15X,'NUMBER OF ITERATIONS',I4,//,5X,'B(J)',10(2X,F10.8))
ISN 0026    120 FORMAT(5X,25I4)
ISN 0027    200 FORMAT(///,15X,'GREEDY ALGORITHM',///,15X,'PHASE 1',///)
ISN 0028    205 FORMAT(5X,25I4)
ISN 0029    210 FORMAT(///,15X,'INITIAL KIT COMPUTATION',///,5X,I4,'  ITEMS IN COMPUTA
           1TATION',///,20X,'KIT 5718',///)
ISN 0030    220 FORMAT(///,15X,'TARGETS',5X,'PERFORMANCE OF 5718 KIT',///,5X,'COST
           1 OF 5718 KIT=$',F15.2,5X,'NORSGOAL=',F10.5,5X,'SDOGOAL=',F10.5,///)
ISN 0031    230 FORMAT(5X,'BUDGET OF INITIAL KIT IS  $ ',F15.2,///)
C
ISN 0032     50 IPHASE=2
ISN 0033        RETURN
ISN 0034        END
```

```
ISN 0002            SUBROUTINE INPUT2(KIT,PRICE)
                    C
                    C   READ IN THE PERTINENT DATA FROM THE TAPES AND CARDS
                    C
ISN 0003            DIMENSION KIT(500),SN(4),TM(4),DPGM(30),DMDS(30),RQMT(30),RPRS(30)
ISN 0004            COMMON PUISON(500,6),BO(5)
ISN 0005            COMMON /BLOCK1/ITOT,NITER,NOFS,NUF,BUDGET,SKF,FAR
ISN 0006            COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007            COMMON /BLOCK4/ KT57,8(500),VGOAL,SGOAL,BUDKT
ISN 0008            COMMON /PHASE/ IPHASE
ISN 0009            COMMON /INPUT/NDAYS,OFP,NSUD
ISN 0010            DOUBLE PRECISION PUISON,BO
ISN 0011            DATA XRU/'LRU'/,SRU/'SRU'/,RRR/'RRR'/
ISN 0012            DATA DPGM/7*1.15,.53,22*.52/
                    C
ISN 0013            NITER=0
ISN 0014            PRICE=0.0
ISN 0015            ITEM=0
                    C
ISN 0016            READ(5,109) PAR
                    C
ISN 0017          10 ITEM=ITEM+1
                    C
ISN 0018          20 READ(1,111,END=200) (SN(I),I=1,4),CAP, NRC,NAP(ITEM),TDR,DDR,COST
                     *(ITEM),TYP
ISN 0019            BRR=TDR-DDR
                    C
                    C   ONLY LRU & SRU ITEMS WILL BE PROCESSED
                    C
ISN 0020            IF(TYP.EQ.SRU) GO TO 30
ISN 0022            IF(TYP.NE.XRU) GO TO 20
ISN 0024            INU=KIT(ITEM)
ISN 0025            GO TO 17
ISN 0026          30 INU=0
ISN 0027          17 TOTRQ=0.0
ISN 0028             DO 110 IDAY=1,NDAYS
ISN 0029          110 DMDS(IDAY)=FLOAT(NAP(ITEM))*TDR*DPGM(IDAY)
                    C
                    C   RRR PROCESSING
                    C
ISN 0030            IF(CAP.NE.RRR) GO TO 15
ISN 0032            IF(TYP.EQ.XRU.AND.INU.LT.1) INU=1
ISN 0034            KTOT=NSUD+NRC
ISN 0035            IF(KTOT.GT.30) KTOT=30
ISN 0037            DO 120 IDAY=1,KTOT
ISN 0038          120 RPRS(IDAY)=0.0
ISN 0039            RPRHOL=0.0
ISN 0040            IF(KTOT.GE.30) GO TO 132
ISN 0042            KTOT=NSUD+1
ISN 0043            DO 125 I=1,KTOT
ISN 0044          125 RPRHOL=RPRHOL+FLOAT(NAP(ITEM))*BRR*DPGM(I)
ISN 0045            RPRS(NSUD+NRC+1)=RPRHOL
ISN 0046            JTOT=NSUD+NRC+2
ISN 0047            DO 130 IDAY=JTOT,NDAYS
ISN 0048          130 RPRS(IDAY)=FLOAT(NAP(ITEM))*ERR*DPGM(IDAY-NRC)
ISN 0049          132 RQMT(1)=DMDS(1)-RPRS(1)
ISN 0050            DO 135 IDAY=2,NDAYS
ISN 0051          135 RQMT(IDAY)=RQMT(IDAY-1)+DMDS(IDAY)-RPRS(IDAY)
ISN 0052            TOTRQ=RQMT(1)
ISN 0053            DO 140 IDAY=2,NDAYS
ISN 0054            IF(RQMT(IDAY).GT.TOTRQ) TOTRQ=RQMT(IDAY)
ISN 0056          140 CONTINUE
ISN 0057            GO TO 70
```

```
                C
                C    RR  PROCESSING
                C
ISN 0058           15 DO 160 IDAY=1,NDAYS
ISN 0059          160 TOTRQ=TOTRQ+DMDS(IDAY)
                C
                C    DEMAND AND KIT  VALUES
                C
ISN 0060           70 XMU(ITEM)=TOTRQ
ISN 0061              KIT(ITEM)=INU
ISN 0062              PRICE=PRICE+COST(ITEM)*FLOAT(INQ)
ISN 0063              GO TO 10
                C
ISN 0064          200 ITOT=ITEM-1
                C
                C         COMPUTATION OF THE REQUIREMENT KIT 5718 & TARGET VALUES
                C
ISN 0065              BUDGET=0.0
ISN 0066              DO 50 I=1,ITOT
ISN 0067              KT5718(I)=IFIX(XMU(I)*SKF+.5)
ISN 0068              IF(KT5718(I).LT.1) KT5718(I)=1
ISN 0070              BUDGET=BUDGET+COST(I)*FLOAT(KT5718(I))
ISN 0071           50 CONTINUE
ISN 0072              BUDKT=BUDGET
                C
ISN 0073          102 FORMAT(5X,4A4,5X,I4)
ISN 0074          105 FORMAT(I4)
ISN 0075          108 FORMAT(1UD8.0)
ISN 0076          109 FORMAT(F10.0)
ISN 0077          111 FORMAT(4A4,T19,A3,1X,I1,1X,I2,T32,F5.4,1X,F5.4,T46,F8.2,A3)
ISN 0078              RETURN
ISN 0079              END

ISN 0002              SUBROUTINE OTPUT2(KIT,PRICE,ENORS,SDO)
                C
                C    THIS SUBROUTINE PRINT THE FINAL RESULTS OF THE OPTIMIZATION
                C
ISN 0003              DIMENSION KIT(500)
ISN 0004              COMMON POISON(500,60),BO(5)
ISN 0005              COMMON /BLOCK1/ITOT,NITER,NOPS,MUF,BUDGET,SKF,FAR
ISN 0006              COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007              COMMON /BLOCK4/ KT5718(500),VGOAL,SGOAL,BUDKT
                C
ISN 0008              DOUBLE PRECISION POISON,BO
                C
ISN 0009              WRITE(6,99)
                C
                C     PRINT OUT THE TARGET VALUES
                C
ISN 0010              WRITE(6,950) VGOAL,SGOAL,BUDKT
ISN 0011              WRITE(6,990) (KT5718(I),I=1,ITOT)
                C
ISN 0012              WRITE(6,50) ITOT,NITER
ISN 0013              WRITE(6,100) ENORS,SDO,PRICE
ISN 0014              WRITE(6,200) (KIT(I),I=1,ITOT)
ISN 0015              WRITE(6,300) (BO(J),J=1,MUF)
ISN 0016              WRITE(6,555) FAR
                C
ISN 0017           50 FORMAT(///,5X,'TOTAL NUMBER OF ITEEMS UNDER CONSIDERATIONS',I4,///,
                     +5X,'NUMBER OF ITERATIONS',I4,///)
ISN 0018           99 FORMAT(///,15X,'GREEDY ALGORITHM',//,15X,'PHASE 2',///)
ISN 0019          100 FORMAT(5X,'E(NORS/K)=',F10.5,5X,'E(SDO/K)=',F1(.5,5X,'PRICE(K)=$',
                     +F15.5,///,15X,'OPTIMAL KIT',///)
ISN 0020          200 FORMAT(5X,25I4)
ISN 0021          300 FORMAT(///,5X,'B',/,(1(2X,F10.8)))
ISN 0022          555 FORMAT(///,5X,'WEIGHT ',F1(.3,///)
ISN 0023          950 FORMAT(///,5X,'TARGETS',///5X,' NORSGOAL ',F10.5,5X,'SDOGOAL=',F10.
                     +5,5X,'PRICE=$',F15.2,///)
ISN 0024          990 FORMAT(5X,25I4)
ISN 0025              RETURN
ISN 0026              END
```

```
ISN 0002            SUBROUTINE TABLE
                C
                C   COMPUTATION OF THE CUMULATIVE SUMS OF POISSON DISTRIBUTON
                C
ISN 0003            COMMON POISON(500,60),BO(5)
ISN 0004            COMMON /BLOCK1/ITOT,NITER,NOPS,MUF,BUDGET,SKF,FAR
ISN 0005            COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
                C
ISN 0006            DOUBLE PRECISION POISON,TERM,XXX,SUM,TERL
                C
ISN 0007            TERL=5.D-12
ISN 0008            DO 10 I=1,ITOT
ISN 0009            XXX=DBLE(XMU(I))
ISN 0010            TERM=DEXP(-XXX)
ISN 0011            POISON(I,1)=TERM
ISN 0012            DO 20 J=2,60
ISN 0013            IF(TERM.LT.TERL.AND.POISON(I,J-1).GT.9.D-01) GO TO 30
ISN 0015            TERM=TERM*XXX/DFLOAT(J-1)
ISN 0016            POISON(I,J)=POISON(I,J-1)+TERM
ISN 0017         20 CONTINUE
ISN 0018            GO TO 10
ISN 0019         30 DO 40 JX=J,60
ISN 0020         40 POISON(I,JX)=1.D+00
ISN 0021         10 CONTINUE
ISN 0022            RETURN
ISN 0023            END
```

```
ISN 0002            SUBROUTINE INKIT(KIT,PRICE)
                C
                C       INCREASE KIT VALUES TO ENSURE THAT THE PROBABILITY OF ANY ITEM
                C   WHICH WILL HAVE HALF OF THE FLEET DOWN IS LESS THAN 50%
                C
ISN 0003            DIMENSION KIT(500)
ISN 0004            COMMON POISON(500,60),BO(5)
ISN 0005            COMMON /BLOCK1/ITOT,NITER,NOPS,MUF,BUDGET,SKF,FAR
ISN 0006            COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007            COMMON /BLOCK3/ KITIN(500),PRCIN
                C
ISN 0008            DOUBLE PRECISION POISON,BO
                C
ISN 0009            NEPS=(NOPS+1)/2
ISN 0010            DO 10 ITEM=1,ITOT
ISN 0011         20 NUM=NEPS*NAP(ITEM)+KIT(ITEM)
ISN 0012            IF(NUM.GT.60) GO TO 10
ISN 0014            IF(POISON(ITEM,NUM).GT..5) GO TO 10
ISN 0016            KIT(ITEM)=KIT(ITEM)+1
ISN 0017            PRICE=PRICE+COST(ITEM)
ISN 0018            GO TO 20
ISN 0019         10 KITIN(ITEM)=KIT(ITEM)
ISN 0020            PRCIN=PRICE
                C
ISN 0021            RETURN
ISN 0022            END
```

```
ISN 0002            SUBROUTINE OPTIM(KIT,PRICE,ERORS,SCO)
                C
                C       MAIN OPTIMIZATION ROUTINE
                C
ISN 0003            DIMENSION KIT(500),S(50)
ISN 0004            COMMON POISON(500,60),BO(50)
ISN 0005            COMMON /BLOCK1/ITOT,NITER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006            COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007            COMMON /BLOCK4/ KT5718(500),VGOAL,SGOAL,BUDKT
ISN 0008            COMMON /PHASE/ IPHASE
                C
ISN 0009            DOUBLE PRECISION POISON,BO,SUMSQ,EPSQ,PRO,EPS,S
ISN 0010            COMMON /S1/S
                C
ISN 0011            DATA EPS,EPSQ/1.D-07,5.D-11/
                C
                C       COMPUTE  THE PERFORMANCE OF KIT 57-18  AND INITIATE PARAMETERS
                C       B
                C
ISN 0012            CALL CALC(KT5718,VGOAL,SGOAL)
ISN 0013            DO 1 J=1,MUP
ISN 0014          1 BO(J)=S(J)
ISN 0015            WRITE(6,300) (BO(J),J=1,MUP)
ISN 0016        300 FORMAT('1'/T9,'INITIAL VALUES OF PARAMETERS B'/ (10(2X,F10.8)))
                C
                C       START OPTIMIZATION
                C
                C       STEP 1
                C
ISN 0017        100 CALL MARGAN(KIT,PRICE)
                C
ISN 0018            NITER=NITER+1
                C
                C       STEP 2
                C       UPDATE THE PARAMETERS B
                C
ISN 0019            SUMSQ=0.D+00
ISN 0020            DO 10 J=1,MUP
ISN 0021            PRO=1.D+00
ISN 0022            DO 20 I=1,ITOT
ISN 0023            KITPJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0024            IF(KITPJ.GT.60) GO TO 20
ISN 0026            PRO=PRO*POISON(I,KITPJ)
ISN 0027            IF(PRO.LT.EPSQ) GO TO 15
ISN 0029         20 CONTINUE
ISN 0030         15 SUMSQ=SUMSQ+((BO(J)-PRO)**2)
ISN 0031            BO(J)=PRO
ISN 0032         10 CONTINUE
                C
                C       STEP 3
                C       COMPARE THE NEW VALUES OF B WITH PREVIOUS VALUES
                C       TERMINATE IF THEY ARE SUFFICIENTLY CLOSE
                C
ISN 0033            IF(SUMSQ.LT.EPS) GO TO 30
ISN 0035            GO TO 100
                C
ISN 0036         30 CALL CALC(KIT,ERORS,SCO)
ISN 0037            RETURN
ISN 0038            END
```

```
ISN 0002            SUBROUTINE MARGAN(KIT,PRICE)
                C
                C       THIS SUBROUTINE PERFORMS THE MARGINAL ANALYSIS
                C
ISN 0003            DIMENSION RATIO(500),KIT(500)
ISN 0004            COMMON POISON(500,60),BO(50)
ISN 0005            COMMON /BLOCK1/ ITOT,NITER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006            COMMON /BLOCK2/ NAP(500),COST(500),XMU(500)
ISN 0007            COMMON /BLOCK3/ KITIN(500),PRCIN
                C
ISN 0008            DOUBLE PRECISION POISON,RATIO,BO,SUM,WW,RMAX
                C
                C
                C       START WITH THE INITIAL KIT AND CALCULATE THE RATIO OF RETURN
                C       PER DOLLAR INVESTED FOR EACH ITEM
                C
ISN 0009            PRICE=PRCIN
ISN 0010            DO 10 I=1,ITOT
ISN 0011            KIT(I)=KITIN(I)
ISN 0012            NMAX1=KIT(I)+(NOPS-1)*NAP(I)+2
ISN 0013            IF(NMAX1.GE.60) NMAX1=60
ISN 0015            IXS=KIT(I)+1
ISN 0016            SUM=DBLE(PAR)*(POISON(I,NMAX1)-POISON(I,IXS))
ISN 0017            DO 20 J=1,MUP
ISN 0018            IXJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0019            IF(IXJ.GE.60) GO TO 21
ISN 0021            WW=POISON(I,IXJ+1)/POISON(I,IXJ)
ISN 0022            GO TO 20
ISN 0023         21 WW=1.D+00
ISN 0024         20 SUM=SUM+BO(J)*DLOG(WW)
ISN 0025         10 RATIO(I)=SUM/DBLE(COST(I))
                C
                C       FIND THE MAXIMUM RATIO
                C
ISN 0026        100 DO 25 I=1,ITOT
ISN 0027            IF(I.EQ.1) GO TO 30
ISN 0029            IF(RATIO(I).LT.RMAX) GO TO 25
ISN 0031         30 IMAX=I
ISN 0032            RMAX=RATIO(I)
ISN 0033         25 CONTINUE
ISN 0034            XPRICE=PRICE+COST(IMAX)
                C
                C       TERMINATE IF THE TARGETS ARE ACHIEVED
                C
ISN 0035            IF(XPRICE.GT.BUDGET) GO TO 50
                C
                C       IF NOT, UPDATE THE KIT VALUE AND THE RATIO
                C
ISN 0037            KIT(IMAX)=KIT(IMAX)+1
ISN 0038            PRICE=XPRICE
ISN 0039            NMAX1=KIT(IMAX)+(NOPS-1)*NAP(IMAX)+2
ISN 0040            IF(NMAX1.GT.60) NMAX1=60
ISN 0042            IXS=KIT(IMAX)+1
ISN 0043            SUM=DBLE(PAR)*(POISON(IMAX,NMAX1)-POISON(IMAX,IXS))
ISN 0044            DO 40 J=1,MUP
ISN 0045            IXJ=KIT(IMAX)+(J-1)*NAP(IMAX)+1
ISN 0046            IF(IXJ.GE.60) GO TO 45
ISN 0048            WW=POISON(IMAX,IXJ+1)/POISON(IMAX,IXJ)
ISN 0049         40 SUM=SUM+BO(J)*DLOG(WW)
ISN 0050         45 RATIO(IMAX)=SUM/DBLE(COST(IMAX))
ISN 0051            GO TO 100
ISN 0052         50 RATIO(IMAX)=0.D+00
                C
ISN 0053            IF(RMAX.GT.0.D+00) GO TO 100
ISN 0055            RETURN
ISN 0056            END
```

```
ISN 0002              SUBROUTINE CALC(KIT,ENORS,SDO)
                  C
                  C       THIS SUBROUTINE COMPUTES THE E(NORS/X) AND E(SDO/X)
                  C
ISN 0003              DIMENSION S(50),KIT(500)
ISN 0004              COMMON POISON(500,60),BO(50)
ISN 0005              COMMON /BLOCK1/ITOT,NITEM,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006              COMMON /BLOCK2/ NAP(50),CCST(500),XMU(500)
ISN 0007              COMMON /S1/S
ISN 0008              DOUBLE PRECISION POISON,S,SUP,VNORS,SDOP,EPS
                  C
                  C
ISN 0009              EPS=5.D-10
                  C
                  C       E(NORS/X) COMPUTATION
                  C
ISN 0010              SUM=0.D+00
ISN 0011              S(NOPS+1)=1.D0
ISN 0012              DO 10 K=1,NOPS
ISN 0013              S(K)=1.D+00
ISN 0014              DO 20 I=1,ITOT
ISN 0015              IXJ=KIT(I)+(K-1)*NAP(I)+1
ISN 0016              IF(IXJ.GT.60) GO TO 20
ISN 0018              S(K)=S(K)*POISON(I,IXJ)
ISN 0019              IF(S(K).LT.EPS) GO TO 15
ISN 0021           20 CONTINUE
ISN 0022              SUM=SUM+S(K)
ISN 0023              GO TO 10
ISN 0024           15 S(K)=0.D+00
ISN 0025           10 CONTINUE
ISN 0026              VNORS=DFLOAT(NOPS)*S(NOPS+1)-SUM
ISN 0027              ENORS=VNORS
                  C
                  C       E(SDO/X)  COMPUTATION
                  C
ISN 0028              SDOP=0.D+00
ISN 0029              DO 50 I=1,ITOT
ISN 0030              JUP=(NOPS-1)*NAP(I)+1
ISN 0031              DO 60 J=1,JUP
ISN 0032              IXJ=KIT(I)+J
ISN 0033              IF(IXJ.GT.60) GO TO 50
ISN 0035           60 SDOP=SDOP+1.D+00-POISON(I,IXJ)
ISN 0036           50 CONTINUE
ISN 0037              SDO=SDOP
ISN 0038              RETURN
ISN 0039              END
```

INITIAL VALUES OF PARAMETERS B

0.00000010 0.00286799 0.06779460 0.25850234 0.48975927 0.68018747 0.81107932 0.89285344 0.94108090 0.96842135
0.98345607 0.99151443 0.99573534 0.99789912 0.99896552 0.99951586 0.99977739 0.99989891 0.99995505 0.99998044
0.99999167 0.99999653 0.99999859 0.99999944 1.00000000

GREEDY ALGORITHM

PHASE 1

INITIAL KIT COMPUTATION

180 ITEMS IN COMPUTATION

KIT 5718

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 3 1 2 1 2 1 16 7 9 9 3 4 1 2 11 1 5 4 6
1 1 1 3 11

TARGETS PERFORMANCE OF 5718 KIT

COST OF 5718 KIT=$ 4039896.00 NORSGOAL= 4.92074 SDOGOAL= 24.49522

BUDGET OF INITIAL KIT IS $ 3635906.00

PRICE(X)=$ 3635861.00 E(NORS/X)= 4.84982 E(SDO/X)= 29.47310

NUMBER OF ITERATIONS 3

B(J) 0.00000000 0.00172540 0.06441003 0.26115723 0.49893585 0.69276720 0.82435903 0.90472555 0.95038868 0.97499286
0.98772442 0.99410652 0.99722461 0.99871559 0.99943528 0.99978806 0.99988454 0.99994995 0.99997868 0.99999168
0.99999634 0.99999852 0.99999942 0.99999977 0.99999991

WEIGHT = 0.023

INITIAL KIT

0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 3 1 0 0 0 1 0 0 1 0 2 2 0 0
1 1 2 0 1 0 2 0 1 0 0 1 1 1 1 3 2 0 0 1 2 0 1 0 1
1 0 1 0 1 0 1 2 2 1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0
0 1 2 1 2 1 0 4 1 0 1 0 0 0 1 1 1 2 0 1 1 1 1 1 0
2 1 1 1 0 0 1 1 1 2 1 1 1 1 1 2 1 0 2 1 0 1 0 0 0
0 0 0 1 0 2 2 0 3 2 1 1 17 6 8 9 4 7 1 3 11 0 6 5 5
1 2 1 3 12

INITIAL VALUES OF PARAMETERS B

0.0 0.0 0.00000001 0.00011449 0.00861967 0.07484124 0.23535679 0.44260779 0.63102230 0.77280281
0.86636330 0.92392780 0.95769561 0.97683397 0.98754667 0.99337298 0.99651327 0.99818555 0.99906608 0.99952462
0.99976077 0.99988171 0.99994153 0.99997163 1.00000000

GREEDY ALGORITHM
PHASE 2

TARGETS

NORSGOAL = 0.13540 SOOGOAL = 163.92674 PRICE=$ 7562983.00

11	1	1	1	1	1	1	1	1	1	1	1	4	3	2	1	2	2	2	1	1	1	1	1	1	
2	2	1	4	2	2	1	1	2	4	13	5	1	1	1	3	3	1	3	1	1	7	5	1	1	
3	2	7	6	1	3	17	1	2	1	1	1	1	4	2	19	13	4	1	1	15	3	3	1	2	
6	3	2	3	8	1	4	4	9	1	4	3	1	15	1	2	1	2	2	3	2	1	1	12	2	
2	1	8	4	7	5	2	9	1	5	1	1	1	1	3	1	2	5	1	4	2	2	2	2	2	
8	18	9	2	1	1	11	2	2	5	1	13	4	12	8	18	25	1	15	1	1	3	21	2	4	
17	6	4	3	2	2	4	1	2	1	2	1	16	7	9	9	3	4	1	2	11	1	5	4	13	
1	1	3	3	11	2	1	1	1	1	1	1	1	1	1	1	1	4	5	1	1	1	1	4	15	
1	3	3	5	2	1	3	5	5	1	7	2	1	2	1	1	2	5	1	3	5	4	1	1	10	
1	10	2	2	2	2	1	4	2	4	2	3	3	3	15	2	2	6	3	2	5	1	4	7	2	
2	1	10	4																						

TOTAL NUMBER OF ITEEMS UNDER CONSIDERATIONS 254

NUMBER OF ITERATIONS 4

E(NORS/XI= 7.15853 E($OO/XI= 36.21477 PRICE(XI=$ 7562944.00

OPTIMAL KIT

10	0	2	6	0	0	0	1	0	0	0	0	6	3	1	1	2	2	5	2	2	2	2	2	2	
3	2	1	3	4	1	2	2	2	4	19	7	1	3	2	2	5	2	4	3	2	10	9	1	1	
4	4	10	0	3	3	20	2	5	1	2	3	2	6	5	24	17	4	1	3	18	4	6	1	4	
7	3	3	3	9	2	7	6	14	4	7	4	2	17	1	2	2	4	3	5	6	2	3	11	3	
2	2	13	5	11	7	3	11	1	3	3	2	2	2	4	3	4	14	2	6	3	3	3	3	3	
12	21	12	3	1	2	13	3	3	5	4	16	6	15	11	23	29	1	18	2	1	6	20	3	4	
18	7	4	5	2	2	2	1	3	2	1	1	17	6	8	9	4	7	1	3	11	1	6	5	11	
1	2	2	3	12	3	0	0	1	0	0	0	1	1	1	1	1	5	7	2	3	0	1	5	19	
2	2	5	9	3	3	6	7	7	1	9	4	5	4	3	2	2	5	0	4	6	4	0	1	11	
2	12	4	3	5	2	3	7	4	7	4	6	4	5	17	4	5	7	5	3	8	2	7	11	2	
3	1	11	6																						

B

0.00000000 0.00000000 0.00000935 0.00376217 0.05835153 0.22116612 0.43810253 0.63279580 0.77581119 0.86953349
0.92667958 0.95987318 0.97850019 0.98868473 0.99413854 0.99700801 0.99849417 0.99925267 0.99963432 0.99982364
0.99991621 0.99996080 0.99998195 0.99999192 0.99999636

WEIGHT = 0.020

APPENDIX C

C.1 Notes on Conversion of the D029 Computer Program to an IBM Compiler

C.2 FORTRAN Listing of the IBM Version of the D029 Algorithm with a Solution Output

APPENDIX C.1

NOTES ON CONVERSION OF THE DO29 COMPUTER PROGRAM

TO AN IBM COMPILER

APPENDIX C.1

NOTES ON THE CONVERSION OF THE CURRENT D029 COMPUTER PROGRAM TO AN IBM COMPILER

IBM and CDC FORTRAN compilers are not exactly the same. Any computer code compatible with a CDC FORTRAN compiler is not compatible with an IBM FORTRAN compiler and vice versa. Because of this, the D029-CDC code has been converted into an IBM compatible code in order that appropriate comparisons could be made between the current D029 program and the Greedy Algorithm program.

CDC FORTRAN compilers accept variable names of seven characters, but an IBM FORTRAN compiler can accept variable names of only six characters. Therefore, variable names with seven characters were truncated to six characters. In addition, the following statements are not accepted by an IBM FORTRAN compiler.

```
IF(IMAIN.EQ.3HRRR). . .
```

and

```
READ(1,107). . .
IF(EOF(1).NE.0) GO TO 200
```

The first statement is converted by defining an alphanumeric variable RRR in the data statement as follows.

```
DATA   RRR/'RRR'/
   .
   .
   .
IF(IMAIN.EQ.RRR). . .
```

The second statement checks for the end of file in the data set from a tape. The conversion is accomplished by combining the two lines together as

```
READ(1,107,END=200). . .
```

The word size of a CDC computer is 64 bits, but on an IBM 370, a word size of 32 bits is utilized. This 64 bit word size causes the memory space on a CDC to be expensive and memory saving measures are frequently devised. Consequently, special subroutines for word packing and unpacking are included in the current D029 program.

It is mentioned on page 11 of the Preliminary Evaluation of D029 WRSK Model [8] that the QPA for each item is packed into six bits or 1/10 of a word. The cumulative probability of demand for an item is packed into 20 bits, or 1/3 of a word. The probability values for quantities K, K+QPA, K+2*QPA, . . ., K+UE*QPA are stored when QPA is greater than one. When a probability value for a quantity between K+j*QPA and K+(j+1)*QPA is needed, this probability is estimated by interpolation. On the IBM 370/168 at Oklahoma State University, sufficient memory is available such that word packing and unpacking is not needed. However, double precision is employed to increase the accuracy in calculating the cumulative probability values for E(NORS/X) and E(SDO/X).

The IBM version of the D029 computer program is summarized as follows. The input section of the current D029 has been adapted as before with some necessary modifications due to machine differences. The optimization routine has not been changed. Subroutine ADD has also been preserved with some modifications. Subroutine TABLE of the IBM version calculates the cumulative Poisson sums and stores the values in the array Poisson (500,60). Poisson sums are retrieved from this array when required. The FORTRAN listing of the IBM version of the current D029 program is provided in Appendix C.2.

APPENDIX C.2

FORTRAN LISTING OF THE IBM VERSION OF THE DO29 ALGORITHM

WITH A SOLUTION OUTPUT

```
C ***********************************************
C *                                             *
C *       0029       ALGORITHM                  *
C *       U S  AIR   FORCE                      *
C *                                             *
C *                                             *
C ***********************************************
ISN 0002        DIMENSION KIT(500)
ISN 0003        DOUBLE PRECISION ENORS,SDO
ISN 0004        COMMON /BLK0/ISUM,IPASS
ISN 0005        COMMON /PRINT/MINPUT,MXMU,MITER
C
C       DETAILED PRINT OUT OPTION. ALL DEFAULT TO ZERO.
C
C       MINPUT=1 WILL PRINT OUT INPUT DATA
C       MXMU=1 WILL PRINT OUT AVERAGE DEMAND XMU(I), AND NFPO(I) FOR EACH
C              ITEM
C       MITER=1 WILL PRINT OUT CPU TIME AND ITEMS ADDED FOR EACH ITERATION
C
C ***
ISN 0006        CALL TIME
ISN 0007      1 CALL INPUT1(    PRICE)
ISN 0008        CALL TABLE
ISN 0009        CALL INKIT(KIT,PRICE)
ISN 0010        CALL OPTIM(KIT,ENORS,SDO,PRICE)
ISN 0011        CALL OTPUT(KIT,ENORS,SDO,PRICE)
ISN 0012        IF(IPASS.EQ.2) GO TO 1
ISN 0014        STOP
ISN 0015        END
```

```
ISN 0002            BLOCK DATA
C
C       BLOCK DATA SUBPROGRAM TO INITIALIZE VARIABLES IN LABELED COMMON
C
ISN 0003            DIMENSION BINTER(10)
ISN 0004            COMMON /BLK6/BINTER/BLK0/ISUM,IPASS
ISN 0005        COMMON /BLK8/VPT,SDOT,VPGOAL,SDOGOL
ISN 0006        COMMON /COUNT/ICOUNT,IADD(200)
ISN 0007        COMMON /PRINT/MINPUT,MXMU,MITER
ISN 0008        DOUBLE PRECISION  VPGOAL,SDOGOL
ISN 0009            DATA BINTER/1.5E6, 3.0E6, 4.5E6,6.0E6, 7.5E6,9.0E6,10.0E6,
               +          3*0.0/,IPASS/1/,VPT,SDOT/2*0./,ISUM/0/,ICOUNT/0/ ,
               +    MINPUT,MXMU,MITER/3*0/
ISN 0010            END
```

```
ISN 0002        SUBROUTINE CALC(KIT,VP,SDO)
C
C          THIS SUBROUTINE COMPUTES THE E(NORS/A) AND E(SDO/A)
C
ISN 0003        DIMENSION S1(500),KIT(500)
ISN 0004        COMMON PJISJN(300,100)
ISN 0005        COMMON /BLK0/ISUM,IPASS
ISN 0006        COMMON /BLK1/ITOT,AUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
               +(500)/BLK3/NUPS,SKF,EUEEFF,IRR
ISN 0007          COMMON /BLK7/S1
ISN 0008        DOUBLE PRECISION PJISJN,S1,VP,SDO ,EPS
ISN 0009        DATA EPS/1.D-12/
```

```
C
ISN 0010        S1(NUP)=1.D+00
ISN 0011        SDU=0.D+00
ISN 0012        DO 10 I=1,ITOT
ISN 0013        JUP=(NOPS-1)*NAP(I)+1
ISN 0014        DO 11 J=1,JUP
ISN 0015        IXJ=KIT(I)+J
ISN 0016        IF(IXJ.GE.NFPO(I)) GO TO 10
ISN 0018     11 SDU=SDU+1.D+00-POISON(I,IXJ)
ISN 0019     10 CONTINUE
C
ISN 0020        VP=0.D+00
ISN 0021        IU=IUEFF
ISN 0022        IKKK=IKK
ISN 0023        DO 25 N2=1,IU
ISN 0024        N=IKKK-N2
ISN 0025        S1(N)=1.D+00
ISN 0026        DO 20 I=1,ITOT
ISN 0027        IXJ=KIT(I)+(N-1)*NAP(I)+1
ISN 0028        IF(IXJ.GE.NFPO(I)) GO TO 20
ISN 0030        S1(N)=S1(N)*POISON(I,IXJ)
ISN 0031        IF(S1(N).LE.EPS) GO TO 22
ISN 0033     20 CONTINUE
ISN 0034     21 VP=VP+FLOAT(N)*(S1(N+1)-S1(N))
ISN 0035        IF(S1(1).EQ.0.D+00) GO TO 30
ISN 0037        GO TO 25
C
ISN 0038     22 S1(N)=0.00
ISN 0039        GO TO 21
ISN 0040     25 CONTINUE
ISN 0041        RETURN
ISN 0042     30 DO 35 N2=1,N
ISN 0043     35 S1(N2)=0.D+00
ISN 0044        RETURN
ISN 0045        END
```

```
ISN 0002        SUBROUTINE TIME
C
C       THIS SUBROUTINE GIVES CPU TIMES
C
C       ELAPSE(I) IS AN ASSEMBLY LANGUAGE SUBROUTINE WHICH TIMES THE
C       EXECUTION OF A PORTION OF THE CALLING PROGRAM
C
ISN 0003        COMMON /BLK0/ISUM,IPASS
ISN 0004        DATA K/0/
ISN 0005        IF(K.EQ.0) GO TO 30
ISN 0007        CALL ELAPSE (ICPU)
ISN 0008        ISUM=ISUM+ICPU
ISN 0009        TCPU= FLOAT(ICPU)/1000.
ISN 0010        TSUM=FLOAT(ISUM)/1000.
ISN 0011        WRITE(6,10) TSUM,TCPU
ISN 0012     10 FORMAT (1X,10('*'),2X,'CPU =',F7.3,' SEC.',5X,'INCREAMENT =',
     +       F7.3,2X,10('*'))
ISN 0013        RETURN
ISN 0014     30 CALL ELAPSE(ICPU)
ISN 0015        K=1
ISN 0016        RETURN
ISN 0017        END
```

```
ISN 0002              SUBROUTINE INPUT1(     PRICE)
                  C
                  C    READ IN THE PERTINENT DATA FROM THE TAPES AND CARDS
                  C
ISN 0003              DIMENSION DPGM(30),DMDS(30),RPRS(30),RUNT(30),NSN(4)
ISN 0004              COMMON /BLK1/ITOT,MUP,BUDGET/BLK2/XMU(500),MAP(500),COST(500),NFPO
                     +(500)/BLK3/NOPS,SKF,IEFF,IKK
ISN 0005              COMMON /BLK0/ISUM,IPASS
ISN 0006              COMMON /BLK12/ITOT1,ITOT2
ISN 0007              COMMON /PRINT/MINPUT,MAMU,MITER
ISN 0008              INTEGER DAYS,MPAI,J,I,RC,MPA
ISN 0009              INTEGER   XRU,RRR,SRU,END
ISN 0010              DATA XRU/'LRU'/,RRR/'RRR'/,DPGM/ 7*1.,15., .53, 22*.52/
ISN 0011              DATA SRU/'SRU'/,END/'END'/
                  C
ISN 0012                1 FORMAT('1'///10('*'),2X,'PHASE ',I1,' CALCULATIONS',2X,10('*'))
ISN 0013              101 FORMAT(    F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0014              102 FORMAT(1X,F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0015              107 FORMAT(4A4,T19,A3,I2,1X,I2,T32,F5.4,1X,F5.4,T46,F8.2,A3)
ISN 0016              177 FORMAT(I5,2X,4A4,2X,A3,I2,1X,I2,2X,F8.4,1X,F8.4,2X,F8.2,2X,A3)
                  C
ISN 0017                  WRITE (6,1) IPASS
ISN 0018              100 IF(IPASS.EQ.1)
                         +READ(5,101) PGMI,DAYS,PGM,NOPS,D1,SKF
ISN 0020                  IF(IPASS.EQ.1)
                         +WRITE(6,102)PGMI,DAYS,PGM,NOPS,D1,SKF
ISN 0022                  MUP=NOPS+1
ISN 0023                  IKK=MUP
ISN 0024                  IUEFF=NOPS
ISN 0025                  ITM=0
ISN 0026                  MM=0
ISN 0027                  IF (D1.GT.DAYS) D1=DAYS
ISN 0029                  PGMI=0.
ISN 0030                  DO 8888 JJ=1,D1
ISN 0031             8888 PGMI=PGMI+DPGM(JJ)
ISN 0032                  IF (IPASS.EQ.1) PGM=PGMI
ISN 0034              103 ITM=ITM+1
                  C
ISN 0035              105 READ (1,107,END=200) NSN,IMAIN,RC,MPAI,TDR,DDR,CUST(ITM),LRU
ISN 0036                  IF (IPASS.EQ.2) GO TO 321
ISN 0038                  MM=MM+1
                  C
ISN 0039                  IF(MINPUT.EQ.1)
                         +WRITE(6,177)MM,NSN,IMAIN,RC,MPAI,TDR,DDR,COST(ITM), LRU
ISN 0041              321 MAP(ITM) = MPAI
ISN 0042                  BRR=TDR-DDR
                  C
                  C    ONLY LRU & SRU ITEMS WILL BE PROCESSED
                  C
ISN 0043                  IF(LRU.LT.XRU) GO TO 105
ISN 0045                  IF (LRU.EQ.XRU) GO TO 108
ISN 0047                  IF(LRU.EQ.SRU) GO TO (105,108),IPASS
ISN 0049                  GO TO 105
ISN 0050              108 TOTRU=J.
ISN 0051                    I1=D1
ISN 0052                  IF (IPASS.EQ.2) GO TO 109
ISN 0054                  IF(IMAIN.EQ.RRR)I1=D1+RC
ISN 0056              109 DO 110 I=1,I1
ISN 0057              110 DMDS(I)=MPAI*TDR*DPGM(I)
ISN 0058                  IF (IPASS.EQ.1) GO TO 145
ISN 0060                  J=D1+1
ISN 0061                  DO 115 I=J,DAYS
ISN 0062              115 DMDS(I)=MPAI*TDR*DPGM(I)
ISN 0063                  IF (IMAIN.NE.RRR) GO TO 155
                  C
                  C      LRU RRR PROCESSING
                  C
```

```
ISN 0065            K=D1+RC
ISN 0066            IF (K.GT.30) K=30
ISN 0068            DO 120 I=1,K
ISN 0069          120 RPRS(I)=0.
ISN 0070            RPRHO=0.
ISN 0071            IF(K.GE.30) GO TO 132
ISN 0073            K=D1+1
ISN 0074            DO 125 I=1,K
ISN 0075          125 RPRHO = RPRHO+ FLOAT(QPAI)*BRR*DPGM(I)
ISN 0076            RPRS(D1+RC+1)= RPRHO
ISN 0077            J=D1+RC+2
ISN 0078            IF(J.GT.30) GO TO 132
ISN 0080            DO 130 I=J,DAYS
ISN 0081          130 RPRS(I)= FLOAT(QPAI)*BRR*DPGM(I-RC)
ISN 0082          132 RQMT(1)= DMDS(1)- RPRS(1)
ISN 0083            DO 135 I=2,DAYS
ISN 0084          135 RQMT(I)=RQMT(I-1) + DMDS(I) - RPRS(I)
ISN 0085            TOTRQ=RQMT(1)
ISN 0086            DO 140 I=2,DAYS
ISN 0087          140 IF (RQMT(I).GT.TOTRQ) TOTRQ=RQMT(I)
ISN 0089            GO TO 165
ISN 0090          145 DO 150 I=1,I1
ISN 0091          150 TOTRQ=TOTRQ+DMDS(I)
                C
                C     RR PROCESSING
                C
ISN 0092            GO TO 165
ISN 0093          155 DO 160 I=1,DAYS
ISN 0094            TOTRQ=TOTRQ+DMDS(I)
ISN 0095          160 IF (IMAIN.NE.RRR) RQMT(I)=TOTRQ
ISN 0097          165 XMU(ITM)= TOTRQ
ISN 0098            GO TO 103
ISN 0099          200 ITOT=ITM-1
                C
ISN 0100            IF(IPASS.EQ.1) ITOT1=ITOT
ISN 0102            IF(IPASS.EQ.2) ITOT2=ITOT
ISN 0104            IF(IPASS.EQ.1) PRICE=0.0
ISN 0106            WRITE(6,193) ITOT
ISN 0107          193 FORMAT(//,5X,'TOTAL NUMBER OF ITEMS UNDER CONSIDERATION IS',I4,//)
ISN 0108            RETURN
ISN 0109            END

ISN 0002            SUBROUTINE OUTPUT(KIT,ENORS,SDD,PRICE)
                C
                C   THIS SUBROUTINE PRINT THE FINAL RESULTS OF THE OPTIMIZATION
                C
ISN 0003            COMMON /BLK1/ ITOT,NUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPD
                   +(500)/BLK3/NOPS,SKF,IJEEFF,IKK/BLK5/LOOPC
ISN 0004            COMMON /BLK0/ISUM ,IPASS
ISN 0005            DIMENSION KIT(500),SN(4)
ISN 0006            DOUBLE PRECISION ENORS,SDD
                C
ISN 0007            ITEN=0
ISN 0008            WRITE(6,100) PRICE,ENORS,SDD
ISN 0009          100 FORMAT(//,5X,'PRICE(A)=$',F15.2,5X,'E(NORS/X)=',F10.5,5X,'E(SDD/X)
                   +=',F10.5,//)
ISN 0010            WRITE(6,110) LOOPC
ISN 0011          110 FORMAT(/ ,5X,'NUMBER OF ITERATIONS IS',I4,//)
                C ***
ISN 0012            CALL TIME
ISN 0013            REWIND 1
ISN 0014            IF (IPASS.EQ.2) STOP
ISN 0016            IPASS=IPASS+1
ISN 0017           50 RETURN
ISN 0018            END
```

```
ISN 0002            SUBROUTINE TABLE
                C
                C   COMPUTATION OF THE CUMULATIVE SUMS OF POISSON DISTRIBUTION
                C
                C     POISON(I,K) = CUMULATIVE PROBABILITY OF DEMANDS OF QUANTITY K-1
                C     FOR ITEM I
                C     FOR K GREATER THAN NFPO(I), POISON(I,K) = 1
                C
ISN 0003            COMMON POISON(300,100)
ISN 0004            COMMON /BLK1/ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
                   +(500)/BLK3/NOPS,SKF,IUEEFF,IKK
ISN 0005            COMMON /BLK0/ISUM,IPASS
ISN 0006            COMMON /PRINT/MINPUT,MXMU,MITER
ISN 0007            DOUBLE PRECISION POISON,TERM,WWW,SUM,TERL
ISN 0008            TERL=5.D-11
ISN 0009            DO 10 I=1,ITOT
ISN 0010            WWW=DBLE(XMU(I))
ISN 0011            TERM=DEXP(-WWW)
ISN 0012            POISON(I,1)=TERM
ISN 0013            DO 20 J=2,100
ISN 0014            IF(TERM.LT.TERL.AND.POISON(I,J-1).GT.9.D-01) GO TO 30
ISN 0016            TERM=TERM*WWW/DFLOAT(J-1)
ISN 0017            POISON(I,J)=POISON(I,J-1)+TERM
ISN 0018         20 CONTINUE
ISN 0019            NFPO(I)=100
ISN 0020            GO TO 10
ISN 0021         30 NFPO(I)=J
ISN 0022            DO 40 JX=J,100
ISN 0023         40 POISON(I,JX)=1.D+00
ISN 0024         10 IF(MXMU.EQ.1)
                   +WRITE(6,*) I,NFPO(I), NAP(I), XMU(I)
ISN 0026            RETURN
ISN 0027            END
```

```
ISN 0002            SUBROUTINE INKIT(KIT,PRICE)
                C
                C        INCREASE KIT VALUES TO ENSURE THAT THE PROBABILITY OF ANY ITEM
                C     WHICH WILL HAVE HALF OF THE FLEET DOWN IS LESS THAN 50%
                C
ISN 0003            DIMENSION KIT(500)  ,KITO(500)
ISN 0004            COMMON POISON(300,100)
ISN 0005            COMMON /BLK1/ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
                   +(500)/BLK3/NOPS,SKF,IUEEFF,IKK/ BLK9/VPZRO,SDOZRO
ISN 0006            COMMON /BLK0/ISUM,IPASS
ISN 0007            COMMON /BLK12/ITOT1,ITOT2
ISN 0008            DOUBLE PRECISION POISON  ,VPZERO,SDOZRO ,VP,SDO
ISN 0009            DATA KITO/500*0/
ISN 0010            IF(IPASS.NE.1) GO TO 3
ISN 0012            DO 1 I=1,ITOT
ISN 0013          1 KIT(I)=0
ISN 0014          2 CALL CALC(KITO,VPZERO,SDOZRO)
ISN 0015              WRITE(6,991)(KITO(IX), IX=1,ITOT)
ISN 0016        991 FORMAT (//15X,     'EMPTY KIT'/(13X,25(2X,I2)))
ISN 0017              WRITE (6,992) VPZERO,SDOZRO,IUEEFF
ISN 0018        992   FORMAT(  / 15X,'VP', F10.5,5X, 'SUO', F10.5,5X,'IUEEFF',I4)
ISN 0019            NEPS=MUP/2
ISN 0020            DO 10 ITEM=1,ITOT
ISN 0021         20 NUM=NEPS*NAP(ITEM)+KIT(ITEM)+1
ISN 0022            IF(NUM.GT.NFPO(ITEM)) GO TO 10
ISN 0024            IF(POISON(ITEM,NUM).GT..5) GO TO 10
ISN 0026            KIT(ITEM)=KIT(ITEM)+1
ISN 0027            PRICE=PRICE+COST(ITEM)
ISN 0028            GO TO 20
ISN 0029         10 CONTINUE
```

```
ISN 0030            CALL CALC(KIT,VP,SDO)
ISN 0031                WRITE (6,91) (  KIT(I), I=1,ITOT)
ISN 0032         91  FORMAT(///15X,'INITIAL KIT',//(3X,25(2X,I2)))
ISN 0033                WRITE (6,994) VP     ,SDO    ,IUEEFF ,PRICE
ISN 0034        994  FORMAT(  / 15X,'VP',  F10.5,5X,  'SDO',F10.5,5X,'IUEEFF',I4,5X,
                    +    'PRICE', F12.2)
ISN 0035            RETURN
ISN 0036          3 IF(ITOT.EQ.ITOT2) GO TO 2
ISN 0038            L1=ITOT+1
ISN 0039            DO 4 M=L1,ITOT2
ISN 0040          4 KIT(M)=0
ISN 0041            GO TO 2
ISN 0042            END



ISN 0002            SUBROUTINE OPTIM(KIT,VP,SDO,PRICE)
                C
                C       MAIN OPTIMIZATION ROUTINE
                C
ISN 0003            DIMENSION KIT(500),KT5718(500)
ISN 0004                DIMENSION BINTER(10)
ISN 0005            COMMON /BLK0/ISUM,IPASS
ISN 0006            COMMON /BLK1/ITOT,NUP,BUDGET/BLK2/AMU(500),NAP(500),COST(500),NFPO
                   +(500)/BLK3/NOPS,SKF,IUEEFF,IKK/BLK4/IFLAG,INT,RATIO/BLK5/LOOPC
ISN 0007                COMMON /BLK6/BINTER /BLK9/VPZERO,SDOZRO
ISN 0008            COMMON /BLK8/ VPT,      VPGOAL,SDOGOL
ISN 0009            COMMON /COUNT/ICOUNT,IADD(200)
ISN 0010            COMMON /PRINT/NINPUT,NAMU,MITER
ISN 0011            DOUBLE PRECISION VP,VPGOAL,VPDELT,VPTEST,SDO,SDOGOL,SDODEL,SDOTES
                   +    ,VPZERO,SDOZRO,VPT.
                C
ISN 0012          990 FORMAT(///  15X,'KIT 5718' // (3X,25(2X,I2)))
ISN 0013          994 FORMAT(T3,'ITERATION',I4,5X,  'VP =',F10.5,  3X,'SDO =',F10.5,  3X,
                   +    'PRICE =',  F12.2,3X,'TOTAL ITEMS ADDED =',I3,/(I7,19(I5))))
ISN 0014          999 FORMAT(5X,'TARGETS TO BE ATTAINED',//,5X,'  COST=$',F10.2,5X,'NORS
                   +GOAL=',F10.5,5X,'SDOGOAL=',F10.5,//)
                C
ISN 0015            IFLAG=0
ISN 0016            LOOPC=0
ISN 0017            BUDGET=0.0
                C
                C       KIT 5718 CALCULATION
                C
ISN 0018            DO 10 ITEM=1,ITOT
ISN 0019            KT5718(ITEM)=IFIX(AMU(ITEM)*SKF+.5)
ISN 0020            IF(KT5718(ITEM).LT.1) KT5718(ITEM)=1
ISN 0022            BUDGET=BUDGET+COST(ITEM)*FLOAT(KT5718(ITEM))
ISN 0023          10 CONTINUE
                C
ISN 0024            CALL CALC(KT5718,VPGOAL,SDOGOL)
ISN 0025            WRITE(6,990) (KT5718(IX),IX=1,ITOT)
ISN 0026            WRITE(6,999) BUDGET,VPGOAL,SDOGOL
ISN 0027            IF (IPASS.EQ.2) VPT=1.1*VPGOAL
ISN 0029            IF( IPASS.EQ.1) BUDGET=BUDGET*.9
                C
ISN 0031            VPDELT=(VPZERO-VPGOAL)/5.0D
ISN 0032            VPTEST=VPZERO-VPDELT
ISN 0033            IVP=0
ISN 0034            SDODEL=(SDOZRO-SDOGOL)/5.0D
ISN 0035            SDOTES=SDOZRO-SDODEL
ISN 0036            VP=VPZERO
ISN 0037            SDO=SDOZRO
ISN 0038            ISDO=1
ISN 0039            IDELTA=1
ISN 0040            GO TO 223
ISN 0041          210 IF( VP.LE.VPTEST) GO TO 220
ISN 0043            IF(SDO.LE.SDOTES) GO TO 222
ISN 0045            LOOPC=LOOPC+1
                C
```

```
ISN 0046          215 CALL ADD(KIT,VP,SDJ,PRICE)
ISN 0047              IF(MITER.EQ.1)
                     +     WRITE(6,994) LJOPC,VP,SDO,PRICE,ICOUNT,(IADD(I),I=1,ICOUNT)
ISN 0049              DO 216 I=1,ICOUNT
ISN 0050          216 IADD(I)=0
ISN 0051              ICOUNT=0
                  C ***
ISN 0052              IF(MITER.EQ.1)
                     +CALL TIME
                  C
ISN 0054              IF(IFLAG.EQ.1) GO TO 350
ISN 0056              GO TO 210
ISN 0057          220 VPTEST=VPTEST-VPDELT
ISN 0058              IVP=IVP+1
ISN 0059              IF(SDO.GT.SDOTES) GO TO 223
ISN 0061          222 SDOTES=SDOTES-SDODEL
                  C
                  C     SELECT PROPER 'RATIO' AND 'INT'
                  C
ISN 0062              ISDO=ISDO+1
ISN 0063          223 IBRAN=6*IVP+ISDO
ISN 0064              IF(IBRAN.GT.36) IBRAN=36
ISN 0066              GO TO (304,305,306,306,306,307,303,304,305,306,306,307,
                     1303,303,304,305,306,307,302,303,303,304,306,307,
                     2302,302,302,302,304,307,301,301,301,301,301,333),IBRAN
ISN 0067          250 GO TO (311,311,311,312,312,312,311,311,312,312,313,313,
                     1311,312,312,313,314,314,312,312,313,313,314,315,
                     2312,313,314,314,315,316,312,313,314,315,316,333),IBRAN
ISN 0068          301 VPTEST=0.0+00
                  C
                  C     'RATIO' AND 'INT' SET TABLE
                  C
ISN 0069          302 RATIO=10.
ISN 0070              GO TO 250
ISN 0071          303 RATIO=25.
ISN 0072              GO TO 250
ISN 0073          304 RATIO=50.
ISN 0074              GO TO 250
ISN 0075          305 RATIO=150.
ISN 0076              GO TO 250
ISN 0077          306 RATIO=500.
ISN 0078              GO TO 250
ISN 0079          307 RATIO=1000.
ISN 0080              SDOTES=0.0+00
ISN 0081              GO TO 250
                  C
ISN 0082          311 INT=ITOT/4+1
ISN 0083              GO TO 210
ISN 0084          312 INT=ITOT/8+1
ISN 0085              GO TO 210
ISN 0086          313 INT=ITOT/16+1
ISN 0087              GO TO 210
ISN 0088          314 INT=ITOT/32+1
ISN 0089              GO TO 210
ISN 0090          315 INT=ITOT/64+1
ISN 0091              GO TO 210
ISN 0092          316 INT=1

ISN 0093              GO TO 210
ISN 0094          333 RATIO=50.
ISN 0095              IF((BUDGET-PRICE).GE.40000.) GO TO 334
ISN 0097              INT=1
ISN 0098              GO TO 215
ISN 0099          334 IF((BUDGET-PRICE).GE.20000.) GO TO 335
ISN 0101              INT=ITOT/64+1
ISN 0102              GO TO 215
ISN 0103          335 IF((BUDGET-PRICE).GE.10000.) GO TO 336
ISN 0105              INT=ITOT/16+1
ISN 0106              GO TO 215
ISN 0107          336 INT=ITOT/8+1
ISN 0108              GO TO 215
ISN 0109          350 IFLAG=0
ISN 0110              RETURN
ISN 0111              END
```

```
ISN 0002            SUBROUTINE ADD(KIT,VP,SDO,PRICE)
                C
                C     THIS SUBROUTINE ADDS INT DIFFERENT ITEMS TO THE KIT BY
                C     MARGINAL ANALYSIS
                C
ISN 0003            DIMENSION TS(500),KIT(500)
ISN 0004                DIMENSION BINTER(10),S1(50),SS1(50),S1DEL(50)
ISN 0005            COMMON POISON(300,100)
ISN 0006            COMMON /BLK0/ISUM,IPASS
ISN 0007            COMMON /BLK1/ITOT,AUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
                   +(500)/BLK3/NOPS,SKF,IDEEFF,IKK/BLK4/IFLAG,INT,RATIO
ISN 0008                COMMON /BLK6/BINTER
ISN 0009            COMMON /BLK7/S1 /BLK8/VPT,      VPGDAL,SDDGDL
ISN 0010            COMMON /COUNT/ICOUNT,IADD(200)
ISN 0011            DOUBLE PRECISION PDISJN,TS,DIF,VP,A,SDO,SDOP,T,
                   +    S1,SS1,S1TEMP,SS1TEM,  S1SUM,  SS1SUM,S1DEL
ISN 0012            DOUBLE PRECISION  VPGDAL,SDOGDL ,EPS ,VPT
ISN 0013             DATA INTER/1/,MFLAG/0/,EPS/1.D-12/
                C
ISN 0014            NOPS1=NOPS+1
ISN 0015            IF(IPASS.EQ.2) BTEMP=BINTER(INTER)
ISN 0017            IF (IPASS.EQ.1) BTEMP=BUDGET
ISN 0019            IF(IPASS.EQ.1.AND.BINTER(INTER).LT.BUDGET) BTEMP=BINTER(INTER)
ISN 0021            S1TEMP=0.DO
                C
ISN 0022             DO 30 JS1=1,NOPS
ISN 0023           30 S1TEMP=S1TEMP+S1(JS1)
ISN 0024            S1SUM=NOPS*S1(NOPS1)-S1TEMP
                C
ISN 0025            DO 99 ITM=1,ITOT
ISN 0026            IXL=KIT(ITM)+1
ISN 0027            IXM=KIT(ITM)+ (NOPS-1)*NAP(ITM)+2
ISN 0028            IF(IXM .GE. NFPO(ITM)) IXM=NFPO(ITM)
ISN 0030            IF(IXL .GE. NFPO(ITM)) IXL=NFPO(ITM)
ISN 0032            DIF= POISON(ITM,IXM)-POISON(ITM,IXL)
                C
ISN 0033            DO 20 K=1,NOPS1
ISN 0034            KJ=KIT(ITM)+(K     -1)*NAP(ITM) +1
ISN 0035            KJ1=KJ+1
ISN 0036            IF (KJ .GE. NFPO(ITM)) GO TO 333
ISN 0038            IF( POISON(ITM,KJ).EQ.0.DO) GO TO 222
ISN 0040            SS1(K     )=S1(K     )*POISON(ITM,KJ1)/POISON(ITM,KJ)
ISN 0041            GO TO 20
ISN 0042          222 SS1(K)=1.DO
ISN 0043            DO 223 KI=1,ITOT
ISN 0044            KI1=KIT(KI)+(K-1)*NAP(KI)+1
ISN 0045            IF( KI.EQ.ITM) KI1=KI1+1
ISN 0047            IF (KI1.GE.NFPO(KI)) GO TO 223
ISN 0049            SS1(K)=SS1(K)*PDISJN(KI ,KI1)
ISN 0050            IF( SS1(K).LE.EPS) GO TO 321
ISN 0052          223 CONTINUE
                C
ISN 0053            GO TO 20
ISN 0054          321 SS1(K)=0.DO
ISN 0055            GO TO 20
ISN 0056          333 SS1(K)=S1(K)
ISN 0057           20 CONTINUE
                C
ISN 0058            SS1TEM=0.DO
ISN 0059            DO 40 JSS1=1,NOPS
ISN 0060           40 SS1TEM=SS1TEM+SS1(JSS1)
ISN 0061            SS1SUM=NOPS*SS1(NOPS1)-SS1TEM
ISN 0062            A=SS1SUM
ISN 0063            TS(ITM)=((VP-A)*DBLE(RATIO)+DIF)/DBLE(COST(ITM))
ISN 0064           99 CONTINUE
                C
```

```
C     ADD 2 OF THE INT4 ITEMS WITH THE HIGHEST IMPROVEMENT RATIOS
C     ADD 1 OF THE NEXT HIGHEST INT-INT/4 ITEMS
C
ISN 0065      INT4=INT/4
ISN 0066      DO 102 I=1,INT
ISN 0067      T=TS(1)
ISN 0068      ITM=1
ISN 0069      DO 101 J=2,ITOT
ISN 0070      IF(TS(J).LT.T) GO TO 101
ISN 0072      T=TS(J)
ISN 0073      ITM=J
ISN 0074    101 CONTINUE
ISN 0075      KIT(ITM)=KIT(ITM)+1
ISN 0076      ICOUNT=ICOUNT+1
ISN 0077      IADD(ICOUNT)=ITM
ISN 0078      IF(MFLAG.EQ.1) CALL CALC(     KIT,VP,SDO)
ISN 0080      PRICE=PRICE+COST(ITM)
ISN 0081      TS(ITM)=J.D+00
ISN 0082      IF(PRICE.GT.BTEMP.OR.(MFLAG.EQ.1.AND.(VP.LE.VPGOAL.AND.
             +    SDO.LE.SDOGOL))    GO TO 105
ISN 0084      IF(INT4.EQ.0) GO TO 102
ISN 0086      KIT(ITM)=KIT(ITM)+1
ISN 0087      ICOUNT=ICOUNT+1
ISN 0088      IADD(ICOUNT)=ITM
ISN 0089      IF(MFLAG.EQ.1) CALL CALC(     KIT,VP,SDO)
ISN 0091      PRICE=PRICE+COST(ITM)
ISN 0092      IF(PRICE.GT.BTEMP.OR.(MFLAG.EQ.1.AND.(VP.LE.VPGOAL.AND.
             +    SDO.LE.SDOGOL))    GO TO 105
ISN 0094      INT4=INT4-1
ISN 0095    102 CONTINUE
C
ISN 0096        CALL CALC(KIT,VP,SDO)
ISN 0097      IF (VP.LE.VPT) MFLAG=1
ISN 0099      RETURN
C
ISN 0100    105 IF((IPASS.EQ.1).AND.(BTEMP.EQ.BUDGET) .OR.
             +                        (MFLAG.EQ.1.AND.(VP.LE.VPGOAL.AND.
             +    SDO.LE.SDOGOL))) IFLAG=1
ISN 0102        CALL CALC(KIT,VP,SDO)
ISN 0103      IF (VP.LE.VPT) MFLAG=1
ISN 0105      IF(IFLAG.NE.1) WRITE(6,995) BTEMP
ISN 0107      IF(IFLAG.EQ.1) WRITE(6,994) PRICE
ISN 0109        WRITE(6,991) (KIT(IX), IX=1,ITOT)
ISN 0110        WRITE (6,993) VP,SDO,IUEFF,PRICE,ITM,COST(ITM)
ISN 0111      S1DEL(1)=S1(1)
ISN 0112      DO 95 J=2,NOPS
ISN 0113     95 S1DEL(J)= S1(J)-S1(J-1)
ISN 0114      WRITE (6,996) (S1DEL(I),I=1,NOPS)
C
ISN 0115      IF(IFLAG.NE.1) INTER=INTER+1
ISN 0117      RETURN
C
ISN 0118    991   FORMAT( /15X,'KIT',//,(3X,25(2X,I2)))
ISN 0119    993   FORMAT(  /15X,'VP',F10.5,5X,'SD',F10.5,5X,'IUEFF',I4,5X,
             +      'PRICE',F12.2,5X,'LAST ITEM ADDED',I5,5X,'ITEM COST=',F10.2,
             +    ///)
ISN 0120    995 FORMAT(/// 2X,'BUDGET $=',F12.2)
ISN 0121    996 FORMAT(5(4X,E20.12))
ISN 0122    994 FORMAT(///2X,'PRICE OF OPTIMAL KIT =',F12.2)
ISN 0123      END
```

```
*********  PHASE 1 CALCULATIONS  *********
 1.10 39 20.02 24        1 1.0000

   TOTAL NUMBER OF ITEMS UNDER CONSIDERATION IS 100


           EMPTY KIT
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0
           VP  17.10638    SDD 139.55135    IUEEFF  24


           INITIAL KIT
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0  4  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0
           VP  14.54962    SDD 135.62115    IUEEFF  24    PRICE  207544.00


           KIT 5718

 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  3  1  2  1  2  1 16  7  9  9  3  4  1  2 11  1  5  4  6
 1  1  1  3 11
TARGETS TO BE ATTAINED
  COST=$4039896.00    NORSGOAL=  4.92374    SDDGOAL=  24.49523
```

BUDGET $= 1500000.00

KIT

0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	1	
1	0	0	0	1	0	1	0	1	0	3	1	0	1	0	0	1	0	1	1	0	2	2	0	0	
1	1	2	0	2	0	3	1	1	0	0	1	1	1	2	3	2	0	1	2	2	1	2	0	1	
1	1	1	0	2	1	1	2	2	1	1	1	0	1	0	0	1	1	1	1	2	0	1	0	1	
0	2	2	1	2	1	1	3	1	0	1	0	1	0	1	2	1	2	0	1	1	1	1	1	1	
2	2	1	1	1	1	1	1	1	2	2	1	1	1	1	2	1	0	2	2	0	1	0	1	1	
0	1	0	1	0	2	0	0	3	3	0	0	10	0	2	4	3	6	1	3	5	0	1	1	0	
1	2	1	0	5																					

VP 10.57887 SDO 72.73502 IUEEFF 24 PRICE 1501613.00 LAST ITEM ADDED 165 ITEM COST= 60485.00

0.0	0.0	0. 114389446716D-08	0.990051649934D-06	0.874508305120D-04
0.182921340822D-02	0. 140900969352D-01	0. 525352360869D-01	0. 114524018955D+00	0. 167535027597D+00
0.182316286315D+00	0. 159550594716D+00	0. 119110209061D+00	0. 792794095890D-01	0. 485936378679D-01
0.280673630137D-01	0. 155208299186D-01	0. 830414776542D-02	0. 432754756073D-02	0. 220530389856D-02
0. 110121340581D-02	0. 539264789139D-03	0. 258990352460D-03	0. 121940974900D-03	

BUDGET $= 3000000.00

KIT

0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1	0	0	1	
1	0	0	0	1	0	1	1	1	1	3	1	0	1	0	0	1	0	1	1	1	2	2	0	0	
1	1	2	0	2	1	3	1	1	0	0	1	1	1	2	3	2	0	1	2	2	1	2	0	1	
1	1	1	0	2	1	1	2	2	1	1	1	0	1	0	0	1	1	1	1	2	0	1	0	1	
0	2	2	1	2	1	1	4	2	0	1	1	1	0	1	2	1	2	0	1	1	1	1	1	1	
2	2	1	1	1	1	1	1	1	2	2	1	1	1	1	2	1	0	2	2	0	1	0	1	1	
0	1	1	1	1	2	0	0	3	3	1	1	15	5	6	8	4	6	2	3	9	0	4	4	4	
2	2	2	1	10																					

VP 6.38619 SDO 35.95377 IUEEFF 24 PRICE 3033970.00 LAST ITEM ADDED 166 ITEM COST= 40102.00

0. 104886787158D-10	0. 299804812750D-04	0. 412166343100D-02	0. 420847808515D-01	0. 129900708552D+00
0. 200863994842D+00	0. 204866061719D+00	0. 161695923971D+00	0. 108994399261D+00	0. 665834445027D-01
0. 381245306590D-01	0. 208404115493D-01	0. 109886532274D-01	0. 562383782597D-02	0. 280525443295D-02
0. 136777634598D-02	0. 653164103593D-03	0. 305674816924D-03	0. 140564619014D-03	0. 634041765937D-04
0. 280688895380D-04	0. 121916890188D-04	0. 519343636375D-05	0. 216875375345D-05	

PRICE OF OPTIMAL AIT = 3672038.00

AIT

2	0	1	2	2	0	2	0	0	2	2	2	1	1	0	0	2	0	1	1	0	1	0	0	1
1	0	2	0	1	0	1	1	1	1	3	1	0	1	0	0	1	0	1	1	1	2	2	2	2
1	1	2	2	2	1	3	1	1	2	2	1	1	1	2	3	2	0	1	2	2	1	2	2	1
1	1	1	0	2	1	1	2	2	1	1	1	0	1	0	0	1	1	1	1	2	0	1	0	1
0	2	2	1	2	1	1	2	2	2	1	1	1	2	1	2	1	2	0	1	1	1	1	1	1
2	2	1	1	1	1	1	1	1	2	2	1	1	1	1	1	2	1	0	2	2	0	1	0	1	1
2	1	1	1	1	2	2	2	3	3	1	1	17	6	8	9	4	6	2	3	11	0	5	5	5
2	2	2	2	12																				

VP 4.90458 SDD 26.04894 IUEEFF 24 PRICE 3672008.00 LAST ITEM ADDED 157 ITEM COST= 41566.00

0.6859679311920-06	0.1925014816270-02	0.5008937071190-04	0.1854552166070+00	0.2393117127510+00
0.2009833778840+00	0.1367924468930+00	0.8282413851330-01	0.4658756997220-01	0.2490446297170-01
0.1281944139680-01	0.6435857981330-02	0.3124116859880-04	0.1492425786360-02	0.7000227716740-03
0.3228529900860-03	0.1465 05 07 57 39 0-03	0.6561239145890-04	0.2872620384210-04	0.1240012736990-04
0.5257524761280-05	0.2187876375600-05	0.8930197723520-06	0.3573253059830-06	

PRICE(AIT)= 3672038.00 E(NORS/XI)= 4.90458 E(SDD/XI)= 26.04894

NUMBER OF ITERATIONS IS 15

********** CPU = 2.561 SEC. INCREMENT = 2.561 **********

********** PHASE 2 CALCULATIONS **********

TOTAL NUMBER OF ITEMS UNDER CONSIDERATION IS 254

EMPTY KIT

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0																					

VP 23.04631 SDD 940.65264 ILEEFF 24

INITIAL KIT

0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1	0	0	1
1	0	0	0	1	0	1	1	1	1	3	1	0	1	0	0	1	0	1	1	1	2	2	0	0
1	1	2	0	2	1	3	1	1	0	0	1	1	1	2	3	2	0	1	2	2	1	2	0	1
1	1	1	0	2	1	1	2	2	1	1	1	0	3	0	0	1	1	1	1	2	0	1	0	1
0	2	2	1	2	1	1	4	2	0	1	1	1	0	1	2	1	2	0	1	1	1	1	1	1
2	6	1	1	1	1	1	1	1	2	2	1	1	1	1	6	13	0	2	2	0	1	9	1	1
5	1	1	1	1	2	2	0	3	3	1	1	17	[illegible]	8	9	4	6	2	3	11	0	5	5	5
2	2	2	2	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
0	0	0	0																					

VP 18.89270 SDD 693.47625 ILEEFF 24 PRICE 4280115.00

KIT 5718

11	1	1	1	1	1	1	1	1	1	1	1	4	3	2	1	2	2	2	1	1	1	1	1	1
2	2	1	4	2	2	1	1	2	4	13	5	1	1	1	3	3	1	3	1	1	7	5	1	1
3	2	7	6	1	3	17	1	2	1	1	1	1	4	2	19	13	4	1	1	15	3	3	1	2
6	3	2	3	8	1	4	4	9	7	4	3	1	15	1	2	1	2	2	3	2	1	1	12	2
2	1	8	4	7	5	2	9	1	5	1	1	1	1	3	1	2	9	1	4	2	2	2	2	2
8	18	9	2	1	1	11	2	2	5	1	13	4	12	8	18	25	1	15	1	1	3	21	2	4
17	6	4	3	2	2	4	1	2	1	2	1	16	7	9	9	3	4	1	2	11	1	5	4	13
1	1	5	3	11	2	1	1	1	1	1	1	1	1	1	1	1	4	5	1	1	1	1	4	15
1	3	3	5	2	1	3	5	5	1	7	2	1	2	1	1	2	5	1	3	5	4	1	1	10
1	10	2	2	2	2	1	4	2	4	2	3	3	3	15	2	2	6	3	2	5	1	4	7	2
2	1	10	4																					

TARGETS TO BE ATTAINED

COST=$7562983.00 MORSGOAL= 8.13540 SOOGOAL= 183.92675

BUDGET $= $500000.01

KIT

0	0	2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	3	1	0	1	0	0	1
1	0	0	0	2	0	2	1	1	1	10	2	0	1	0	0	2	0	1	2	1	7	6	0	0
2	2	7	0	4	1	6	1	3	0	0	2	2	2	4	6	7	9	2	4	3	1	4	0	3
1	1	1	0	2	1	2	6	5	2	4	1	0	5	0	0	1	2	1	1	4	0	2	0	1
0	4	8	1	9	2	1	4	2	0	2	1	1	0	1	4	2	7	0	1	1	1	1	2	1
6	9	1	2	2	1	2	1	2	6	4	5	3	3	1	11	13	0	2	4	0	4	9	1	1
5	1	1	2	1	2	2	0	3	3	1	1	17	6	8	9	4	6	2	3	11	0	5	5	5
3	2	2	2	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	7
1	3	0	5	2	2	3	2	2	1	0	2	1	3	2	1	3	0	0	0	0	0	1	1	0
0	1	1	0	4	0	2	4	3	5	2	4	0	2	5	3	3	0	3	1	4	0	5	6	0
1	0	0	3																					

$P 17.78052 SOO 527.11206 EUEEFF 24 PRICE $302972.00 LAST ITEM ADDED 132 ITEM COST= 5000.00

0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.1415611837580-08	0.4314340456990-06
0.2820916910510-04	0.5858448641710-03	0.5103934d18470-02	0.2283451426690-01	0.6110075108640-01
0.1101293712560+00	0.1469871423380+00	0.1566732232800+00	0.1415393317650+00	0.1134374855400+00
0.8344330437690-01	0.5772942744760-01	0.3820600950700-01	0.2446208112960-01	

BUDGET $= $000000.00

KIT

6	0	2	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	4	1	2	1	1	1	1
2	0	0	0	3	0	2	1	1	1	15	5	0	2	1	0	3	1	2	2	1	9	7	0	0
3	3	9	0	4	1	17	3	4	0	1	2	2	5	4	20	14	0	2	4	14	3	5	0	3
4	1	2	0	5	1	4	7	10	3	5	2	1	13	0	0	1	3	2	2	5	1	2	6	2
0	4	10	3	13	5	1	6	2	0	2	1	2	1	2	4	3	10	2	4	2	2	2	2	1
17	17	7	2	2	2	8	3	2	7	4	12	4	10	6	18	24	0	13	4	0	5	15	2	1
14	2	1	4	1	2	2	0	3	3	1	1	17	6	8	9	4	6	2	3	11	3	5	5	6
3	2	2	2	12	2	0	0	0	0	0	0	0	0	0	0	1	0	3	1	2	0	0	4	14
1	4	3	8	4	2	4	5	5	2	6	3	1	4	3	1	3	2	0	2	3	0	1	1	6
0	9	2	0	4	1	3	5	3	6	3	6	3	4	13	3	4	3	4	2	6	1	7	9	0
2	0	5	4																					

```
          VP  11.16119     SDD 262.59253     IUEEFF  24     PRICE  6301423.00     LAST ITEM ADDED  127     ITEM COST=  5934.00

 0.0                       0.0                       0.0                       0.11450791034 1D-11       0.8824341792100-07
 0.5040 5 7880200-04       0.2546427071203-02        0.2375251678090-11        0.8212259352350-11        0.1503102960440+00
 0.1830833 3590D+00        0.1717325122503+00        0.1359226938920+00        0.9607731129080-01        0.6283781068000-11
 0.3899028993550-11        0.2310878872130-01        0.1330946063450-01        0.7476917218850-02        0.4113923495260-02
 0.2224799 1226D-02        0.1184200256700-02        0.6781067095700-03        0.3181139387240-03

PRICE OF OPTIMAL KIT =  7007734.00

          KIT
 9   0   2   0   0   0   0   0   0   3   0   0   5   3   0   1   1   1   5   2   2   1   1   1   1
 2   1   2   1   5   0   2   2   3   3  17   6   1   2   1   0   4   1   3   2   2  11   8   1   1
 4   4  10   0   4   2  19   3   6   1   1   2   2   5   4  22  16   2   2   4  16   3   5   0   4
 6   2   3   2   7   1   6   7  12   3   6   3   1  16   0   2   3   3   2   4   5   1   2   9   3
 1   4  12   4  11   6   2  10   2   2   3   2   2   1   3   4   3  13   2   5   2   3   4   3   3
12  20  11   3   2   2  11   3   4   8   4  14   6  13   9  22  28   0  16   4   1   7  19   3   3
17   5   2   4   2   2   2   9   3   3   1   1  17   6   8   9   4   6   2   3  11   0   5   5  10
 3   2   2   2  12   2   0   0   0   9   9   0   0   9   0   2   1   4   5   2   2   0   0   5  17
 2   5   5   8   4   3   5   7   6   2   5   3   1   4   3   2   1   4   0   3   6   3   1   1  19
 1  11   3   2   5   2   3   6   5   7   3   6   3   4  16   4   4   5   5   2   7   1   7  10   2
 3   1  10   5

          VP   8.13031     SDD 145.30523     IUEEFF  24     PRICE  7007734.00     LAST ITEM ADDED  108     ITEM COST=  4000.00

 0.0                       0.0                       0.2835129037270-08        0.6720280350570-04        0.6793818177480-02
 0.6226699492000-21        0.1613884592800+00        0.2114527057590+00        0.1919361768240+00        0.1421204515240+00
 0.9344855471460-01        0.5709376153470-01        0.3322321900770-01        0.1866992723500-01        0.1021422428940-01
 0.5467679037561D-02       0.2873079510 00 3-02      0.1495161230320-02        0.7562701400810-03        0.3796565176930-03
 0.1879434012080-03        0.9174488725760-04        0.4414369435440-04        0.2092571668530-04

 PRICE(1)=5     7007734.00     E(NONS/KIT=  8.13031     E(SDD/KIT= 145.30523

 NUMBER OF ITERATIONS IS  92

********** CPU = 24.789 SEC.     INCREMENT = 22.223 **********
```

APPENDIX D

D.1 Listing of F4D Input Data

D.2 Listing of Phase 1 and Phase 2 F4D Input Data

APPENDIX D.1

LISTING OF

F4D INPUT DATA

DO29 SYSTEM CONTROL/INPUT CARD FORMAT

I. "A" CARD : PROGRAM CONTROL CARD

Field	Column	Type *	Description
1	1 - 5	I5	Initial Program in Flying Hours.
2	7 - 8	I2	WRSK Support Period in Days.
3	10 -14	I5	Total Program in Flying Hours.
4	16 -17	I2	Unit of Equipment.
5	19 -20	A2	Run Sequence Number.
6	22 -23	I2	Number of Kit Authorizations.
7	25 -26	I2	Number of Set Up Days.
33= STOCKAGE FACTOR ~~8~~ 7	54	A1	A (Card Indicator).

II. "B" CARD : KIT SUMMARY CONTROL CARD

1	1 -10	A10	Kit Serial Number.
	54	A1	B (Card Identifier).

III. "C" CARD : ITEM DATA CARD

1	1 -15	A15	National Stock Number (Left Justified).
2	17	I1	Note Code.
3	19 -21	A3	RR/RRR Maintenance Concept (Left Justified).
4	22 -23	I2	Repair Cycle Time in Days.
5	25 -26	I2	Quantity Per Assembly.
6	28 -30	A3	Application Code.
7	32 -36	F5.4	Total OIM Demand Rate.
8	38 -42	F5.4	Depot Demand Rate.
9	44	A1	Routing Identifier.
10	46 -53	F8.2	Unit Cost.
11	54 -56	A3	LRU/SRU/EOQ/NOP Type Item.
12	58 -62	A5	Work Unit Code (Left Justified).
13	73 -80	A8	Noun.

* Type	Explanation
I	Numeric input only, leading zeros as required, right justified.
A	Alphabetic or numeric input.
F	Numeric Input, decimal assigned by system, leading zeros as required, right justified.

```
        00000000011111111112222222222333333333344444444445555555556666666667777777778
        12345678901234567890123456789012345678901234567890123456789012345678901234567890
CARD
0001    12700CC6S4117    2 RR CO 01 F4D C5689 03072 L 02S927COLRL 74E40          CDU
0002    12700010S5653    1 RR 00 01 F4D 00168 00143 L 0030S4CCLRU 73EBC         CTL SET
0003    12700010S5737    1 RR CO 01 F4D 00325 00094 L 00C6CCCCLRL 73ICC         IND
0004    12700034E1S5E    1 RR CO 01 F4D CC418 C0063 I CCE1EECCLRL 74CF0         PWR SUP
0005    12700C3462C91    1 RR 00 01 F4D 00250 CCC3E L C111C7CCLRL 73CB0         HOUSNG
0006    12700C34S8215    1 RR 00 01 F4D 00325 00049 L 00316200LRU 73CA0         IND
0007    12700034S5637    1 RR 00 01 F4D CC111 C0C17 L C139CCCCLRL 73C00         SRVC
0008    12700035S2209    1 RR 00 01 F4D 00350 C0C7C L 0023320CLRU 73CW0         PWR SUP
0009    12700C3S39141    1 RR CO 01 F4D C0313 C0159 L 00286800LRU 73CD0         CTL AY
0010    12700055184S1    1 RR 00 01 F4D C0313 00281 L 0035ECCCLRL 73E7T         SRVC
0011    12700C9C51639    2 RR 00 01 F4D 00639 00639 L 00E7E4COLRL 74E10         GYROSCOP
0012    12700102284E11   1 RR CO 01 F4D C0350 00021 L 0123660OLRU 73CF0         LSR CTL
0013    12800C9338792NT  2 RR 00 01 F4D 02222 00667 H CC1627CCLRL 731EC         CONF
0014    12800C9338793NT  2 RR 00 01 F4D 01428 01043 F 0023ESCOLRU 731D0         COMP
0015    14300C5128486BF  2 RR CO 01 F4D CC591 00591 G 0CE2EECCLRL 76CAF         SCT AY
0016    14300C6912439JB  1 RR 00 01 F4C 00281 00143 G CC132ECCLRL 74AFS         CTL SEL
0017    1560CC11S2SEEBF  2 RR 00 01 F4C 01051 00855 G C02S7500LRU 4321A         COMP
0018    15600011S2S69EF  2 RR 00 01 F4D CC546 00755 G CC252CCCLRL 4153G         DUCT AY
0019    1560002243897EF  2 RR 00 01 F4C 01137 00412 G CCC3C324LRL 76C9S         RAD CHIN
0020    1560CC7SCE502BF  2 RR CO C1 F4C 00439 C0258 G C0C7S900LRU 1433E         EELL AY
0021    156J0091936978F  2 RR CO 01 F4D 00552 C0403 G CC1424CCLRL 4152G         DUCT AY
0022    1560C0919369SEF  2 RR 00 01 F4C 00502 00336 G 0C1CEECCLRL 4152H         DUCT AY
0023    1560C0954774EBF  2 RR CO C1 F4D C0350 00252 G 00CE4500LRU 4123G         DUCT AY
0024    1560C0954775ZEF  2 RR CO 01 F4D C0411 00353 G CCCS15ECLRL 4123G         DUCT AY
C025    1620CCCE82C64    2 RR 00 01 F4C 00326 00291 G 00C7177CLRL 1334H         VALVE
C026    1620CC943E74S    2 RR 00 01 F4D CCE28 007EE G CCCS27CCLRL 1334C         CTL EXST
0027    1620C0985S1992   2 RR 00 01 F4C CC889 00883 G 0C3122CCLRL 1334B         PWR LT
CC2E    163CCC75E4784    2 RR CO 01 F4C 00526 00491 G 002122C0LRU 1351M         VALVE
0029    16500C00E5895BF  2 RR 00 01 F4D C1E55 C1762 G CCE6S1CCLRL 1432F         STAE CYL
C030    1650C03500992EF  2 RR 00 01 F4C C1006 C0919 G CCCSE2CCLRL 4521C         SEPARATC
C031    1650C03E1E726BF  2 RR CO 02 F4C C0476 C0462 C C0374500LRU 1422E         CYL AY
0032    16500C875C693BF  2 RR CO 01 F4C CC452 CC184 G CCCE527CLRL 1442D         CMF RUCR
C033    1650CC9243005EF  2 RR 00 01 F4C 00642 00422 G 0013330OLRU 1233P         CAMPER
0034    1650009243C06BF  2 RR CO C1 F4D CC773 C0520 G 0C133300LRL 1236N         CAMPER
0035    1660001359566    2 RR 00 01 F4D C1987 01947 H CC32C2CCLRL 41128         TURBINE
C036    1660CC1S52725    2 RR 00 01 F4C 06250 05750 H 00CS16CCLRL 47210         REG GXY
0037    1660CC7535795    2 RR CO C1 F4D C2522 02421 H CC1241CCLRL 4112H         REG
0038    1660CC7936951    2 RR 00 01 F4C 00500 C048C H CC21CECCLRL 4114G         TURBINE
CC39    1660CC7941333    2 RR 00 01 F4C 00680 00634 H 00082450LRU 4121F         REG
0040    1660CC90S1473    2 RR 00 C1 F4D CC523 C0492 H CC1342CCLRL 4114K         REG
C041    284000867062OFL  2 RR 00 02 F4C CC760 C0350 H CC3914CCLRL 2371A         TANK
C042    1680CC3325E62LS  2 RR CO 02 F4C CC749 C0625 F 00C89173LRU 1226E         INERREEL
0043    1680004500572BF  2 RR 00 C2 F4C C0204 C0182 G CCCS33ECLRL 1433A         ACT
C044    1680CC45CC573EF  2 RR 00 02 F4C 00841 C0800 G 002CEECCLRU 1433A         ACT
0045    1680C70249382LS  2 RR CO C2 F4D C0211 00205 F CC034910LRU 1226V         SEAT REL
0046    29150008334S2PL  2 RR 00 02 F4C 00301 C028C H CC1242CCLRL 2353R         AMP
C047    29150C11CSS14FL  2 RR CO 20 F4C 00166 00157 H 00C4717CLRL 2361J         NOZZLE
C048    2915C012SE617PL  2 RR CO 05 F4D C0476 C0447 H CCC3C43CLRL 4611C         VALVE
C049    2915002321350FL  2 RR 00 01 F4C 00382 00363 H CC1317CCLRL 2361C         TCRCESTR
C050    2S15CCSS9119SUF  2 RR 00 02 F4C 00255 00230 G 0C20E3COLRL 4614E         PMPG ELE
C051    29350078S2422    2 RR CO 02 F4D CCE74 0CE2E H CC1C44CCLRL 2371G         CCCLR AY
C052    2S950C0841115FL  2 RR 00 08 F4C 00143 00133 H CCC637SOLRL 2353I         ACT
C053    2S95C015SE730    2 RR CO 02 F4C C1666 01616 H 0004E7SOLRL 2396C         THRTL EX
```

```
      CCOCCCCCCCIIIIIIIIIIZZZZZZZZZZZ3333333333444444444455555555556666666666777777777788
      1234567890123456789012345678901234567890123456789012345678901234567890123456789012
CARD
C054  2595CCCS11224    2 RR CO C2 F4D C1426 C1312 F C0S90100LRU 2381M          STRTR
0055  29950079E2598FL  2 RR CO 02 F4C CC288 00245 H CCC2753CLRL 2322J          CTL EX
C056  4310C1013C4CEF  2 RR 00 01 F4C 01482 01394 G 0025782CCLRL 4221A          COMP
0057  4320C00SEES25HS  2 RR CO C4 F4D C2095 01152 F 00113000LRU 4511E          PMP AY
C058  4810007631105EF  2 RR 00 01 F4C 00663 00558 G CCCE8ECCLRL 4183E          VLV ELC
C059  48100CS624394EF  2 RR 00 05 F4D 00245 00233 C 00C32030LRU 4621A          REG VLV
C060  4820001E54S94BF  2 RR CO C1 F4D C0363 00345 C CC13CCCCLRL 4152C          VLV ELC
C061  4820CC4977437EF  2 RR 00 01 F4C 00484 00466 G CC13CCCCLRL 4152D          VLV BLC
C062  482CCCE6S1SOCBF  2 RR CO 01 F4C CC537 CC345 C 000456S0LRU 42499          SEL VLV
C063  48200C9SS2466YQ  2 RR 00 01 F4D CC377 C026C F CCC3CS70LRL 4123H          VLV
CC64  58260C08S7912    2 RR 00 01 F4C 01782 C1622 L GGCS4550LRU 71LJG          IDEE3C
C065  5865CC16E3S6EEG  2 RR CO C1 F4D C1054 CCCSS L C0028800LRU 76C50          CET
0066  5865001EE3SG7EW  2 RR CO 02 F4C C4766 C2345 L CGCS7ECCLRL 76C4C          CET
C067  5865C016E3S68EW  2 RR 00 02 F4C 03150 00126 L 00C7CE00LRL 76CEG          AMP
C068  5865001EE3572CW  1 RR CO C1 F4D C1775 C0358 L C04GC000LRU 76C90          SIG PROC
0069  5895008257334    2 RR 00 02 F4C 00C45 C0C39 L CCC4122CLRL 71LKC          IND
C07C  5S30CCE47CC37FL  2 RR CO 02 F4C 00304 00283 F 00C2E210LRL 2383E          SWTCH
C071  61100CCS7E394BF  2 RR CO 02 F4D C2657 03476 G CC13SECCLRL 42E40          PANEL
C072  611000571765 4EF 2 RR 00 01 F4C 01503 01358 G CC1E74CCLRL 4223C          CTL EX
CC73  6605CC1113E45    2 RR CO 01 F4D 01428 01256 F 00C28000LRL 731L0          ML1 TRNS
0074  6610NC10S952F    4 RR 00 C1 F4D CCSSS CC471 H CC3ESECCLRL 512CA          CCNF
CC75  661000C0E572766F 2 RR 00 01 F4C CC843 00622 G C0C33SCCLRL 511AD          IND
C076  66100C0863E40    2 RR CO 02 F4C C1619 01457 F 00212900LRU 511AJ          ALT
0077  6610001SC67ES     2 RR CO 01 F4C C1302 011EE F CC21CECCLRL 732AD          IND
C07E  661CCC4CC1202EF  2 RR 00 01 F4C 01112 01112 G 00115800LRU 513EC          IND
C079  66100C7S9E31S    2 RR CO C1 F4D C1674 01456 F 00355000LRU 512CL          IND
0080  6610008491C70    2 RR 00 01 F4C C3782 03328 F CC325CCCLRL 731HC          IND
C081  6610CCEE2SS01EF  2 RR CO 01 F4C 00410 00357 G C0CEC240LRL 511AE          IND
C082  6610CC8831034    2 RR CO C1 F4D C2208 C2031 H CC1115CCLRL 71E10          IND
C083  6610CC9242E27    2 RR 00 02 F4C 01000 C091C F CCC2E7ECLRL 511AA          ACCEL
C084  6610CC9493112BF  2 RR CO 01 F4C 04729 04303 G 00105100LRU 513E0          XMTR AGA
C085  661000953S670    2 RR 00 01 F4C C1176 C1C53 F CCCE755CLRL 511AF          IND
CC86  661000C98E7628EF 2 RR 00 02 F4C 01C86 01C86 G CCC7E65CLRL 511AB          IND
C087  6610CC9942217C   2 RR CO 01 F4C C1291 C111C F 00140100LRU 71E30          IND
0088  66100103476 16   2 RR CO 01 F4C CC547 C0284 F CC15ESCCLRL 512CM          AMP HCR
C089  6615CCCES3851    2 RR 00 01 F4C 07359 06550 F 00618000LRL 731G0          GYRO
C090  6615C037352548F  2 RR CO C1 F4D C0451 C0437 G 00255300LRU 1442F          CYL
C091  661500420C0406EF 2 RR CO 02 F4D C0509 C0485 G CC225ZCCLRL 1425A          ACTUATR
C092  6615CCECCCS69BF  2 RR 00 01 F4C 00494 00494 G 00CE312CLRL 52250          ACCEL
C093  6615CC6CC1CC7BF  2 RR CO C1 F4D C1012 C1012 G CCCE41ESLRL 5224C          ACCEL
C094  6615CC72C2931    2 RR 00 01 F4C 01C00 C0920 F CC1421CCLRL 731KC          XMTR
C095  6615CC7S4E285BF  2 RR 00 01 F4D 01660 01660 C 00165600LRU 52280          RATE GYR
C096  6615007SC1466BF  2 RR 00 C1 F4C C1180 C11EC G CCC1ESCCLRL 52290          GYRC
CC97  661S0C79E1467BF  2 RR 00 01 F4C 00609 00597 C 00165ECCLRU 52270          GYRCSCOP
C098  6615CCS0E4E84    2 RR CO C1 F4C CC553 00138 F 00C59120LRU 731F0          CTLR
C099  661501015S539BF  2 RR 00 C1 F4C C5778 CC173 G C21ESECCLRU 522EC          CTL AMP
C100  662006085E225    2 RR CO 02 F4D CC528 CC457 F CC1275C0LRL 2394A          XMTR
C101  6645CC07C0320    2 RR 00 01 F4D 00767 C0605 P C02CEECCLRL 73FAJ          TIMER
C102  66450CC7C305C    2 RR 00 01 F4C 00328 00231 F 00025360LRU 512AA          CLCCK
C103  6680006S18045    2 RR CO 01 F4D C4225 C2651 F CCCECSE4LRL 47210          INC SET
C104  66800C8945005EF  2 RR 00 01 F4C 01886 01720 G 0013C8CCLRU 46420          IND
C105  66850CE8E5176    2 RR CO C2 F4C C1818 01708 F 00C45730LRU 2392A          INC EGT
0106  672000S8C7854    2 RR 00 C1 F4D C2363 C0354 G CC1224CCLRL 77J2B          CANKD26B
C107  676CCCE747S4E    2 RR CO 01 F4C 00932 00078 G CC13ESCCLRL 77J10          CTL BX
C108  1430000206115BF  2 RRRC3 01 F4D C5325 C45EC G CC4CCCCCLRL 74310          TRAN RCR
```

```
      0000000011111111112222222222333333333344444444445555555556666666667777777778
      1234567890123456789012345678901234567890123456789012345678901234567890123456789
CARD
0109  14300101320485BF 2 RRRC3 01 F4D C1394 00014 G C02SECCCLRL 74770          CTL
C110  1270009375051    2 RR 00 01 F4C 02410 00530 L C1238900LRL 74820          ANF
C111  1650CC9533832BF 2 RR C0 C2 F4C CC349 00058 G CCC430S0LRU 1425E          SPLR CYL
0112  16500C9548231EF 2 RR 00 01 F4D CCSSS CC167 G CC1CE2CCLRL 1351A          CYL AY
C113  1650CC995S494EF 2 RR 00 01 F4C 00668 00260 C CCCES2CCLRL 1234K          CYL
0114  16500CC7S0S84EF 2 RR 00 02 F4D C0261 00034 G 00142900LRU 14258          CYL SPLR
0115  1650003777056EF 2 RR 00 02 F4C 00734 C0465 G CC14CCCCLRL 1455E          CYL FLF
C116  1650CC4012556EF 2 RR 00 02 F4C 00266 00085 G 0002500CLRU 1455F          CYL FLP
0117  16500C7SCEESSBF 2 RR C0 01 F4D CCE83 C042C G CCC4717CLRL 1233K          CYL ACT
C118  1660000714255    2 RR 00 01 F4C 04486 02333 F C0C7157CLRL 471AA          LCX CCNV
C119  1660CCCCE53553   2 RR 00 01 F4C CC704 C0556 F 00166600LRU 4112E          VALVE
0120  1660004959012BF 2 RR 00 01 F4D C2174 01EC4 G CC144SCCLRL 4111M          TMF CTL
C121  1680CC7335768LS 2 RR 00 02 F4C 00399 00331 F 0CCS70S0LRL 1226J          ACT SEAT
C122  4810000685355CTP 2 RR C0 01 F4D C1111 00522 F 00127800LRU 4114J          VALVE
0123  4810007631104EF 2 RR 00 01 F4D CCS72 CC523 G CCCEEECCLRL 41S3E          VLV ELC
C124  4820000345S5EEF 2 RR 00 03 F4C 00258 00154 G 0CCE5S4CLRU 1455C          FLC DVDR
0125  5626CC845CC55   2 RR 00 01 F4D CC751 CC112 L 00111900LRU 71LM0          AS909A
0126  5831001346157CW 2 RR 00 02 F4C C2101 C0084 L CCC7C2ECLRL 71LNC          INTF CCM
C127  5841CC0656743    2 RR 00 01 F4D 08926 01428 L 00595400LRU 723AC          RCVR TRN
C128  5841OC073E241    2 RR C0 01 F4D C4592 C1C5E L CC23ECCCLRL 723E0          IND HGT
C129  5865C016E3564EW 2 RR 00 01 F4C CC784 C0192 L CCCE5CCCLRL 76DBC          PWR SLP
C130  5E6EC01E83565EW 2 RR 00 01 F4C 0C050 CC005 L CCC4E000LRU 76CC0          ANAL
0131  5865001683571EW 2 RR 00 01 F4D CCE41 00231 L CCC84CCCLRL 76DD0          IND
C132  5E65CC3866737EW 2 RR 00 01 F4C C5455 C0491 L CC5CCOCCLRL 76D9C          SIG PROC
C133  5E65CC437E5E3EW 2 RR C0 C1 F4D CC532 C068C F 00101000LRU 76DM0          DET
C134  58650052E7102EW 2 RR 00 01 F4D CCSC7 C04CE L CCC71E75LRL 76DFC          ANF
C135  5865CC997E3E23EW 2 RR 00 02 F4C 01166 00816 L 00C275CCLRL 76D70          GEN SIG
0136  5E65CC597E3E24EW 2 RR C0 01 F4D CCE72 00435 L C0012900LRU 76D60          GEN
C137  5895001EE8798    2 RR 00 01 F4C CE250 C05E2 L CC3CS7CCLRL 71L9C          CCDER
C138  5E550C75S4530    3 RR 00 02 F4D 00884 00062 L 00CE5ECCLRU 71LFC          CTL
0139  5E95CC79CE7E4    2 RR C0 01 F4D CECS4 CCECS L CC3971CCLRL 71LE0          ANF
C140  5895CC81C0136    3 RR 00 01 F4D 04199 C0294 L CC27ECCCLRL 71SDC          SYNCH
C141  5E55CC81CC140    2 RR 00 01 F4D 09050 00814 L 0C271100LRU 71SC0          SW AMPAY
C142  5895008 1C0189   2 RR 00 01 F4D 12658 02666 L CC5E74CCLRL 71SB0          RCVR TRN
C143  5E55CC9113055    2 RR 00 01 F4C 0C706 C0332 L CC3275CCLRL 72S10          TRNSFNDR
0144  5E95CC5150400    1 RR C0 02 F4C C3823 00459 L 00161500LRU 71NE0          CTL
0145  5895009150410    2 RR C0 01 F4D C0337 CCC7E L CCC23E1CLRL 71NFC          CTL
C146  61100C1871018EF 2 RR C0 01 F4C 00401 00196 G 001674CCLRU 42280          CTL BX
C147  63400C11E59E3BF 2 RR C0 02 F4D CCE74 C0762 G CUC43600LRL 4411K          CTL PNL
0148  6605009458168    2 RR C0 01 F4C 10368 04577 M C3551ICCLRL 71E6C          LN12
C149  66C50C5E76166    2 RR C0 01 F4C 01127 00192 F 00147200LRU 71610          CTL PNL
0150  6610004851943EBF 2 RR C0 01 F4D C1812 C1214 G CC273CCCLRL 513X0          ALT ENCO
0151  661000827438SEF 2 RR 00 01 F4C CE693 00695 G C176S4CCLRL 512JC          CADC
C152  6615CCC22EC11    2 RR C0 01 F4D 02979 00864 F 0035140CLRL 731C0          ADAPTR
0153  6615005965172BF 2 RR C0 C1 F4D C1795 CCC5C G CC3223CCLRL 522B0          TRANSDUC
C154  6615CC9825301    2 RR 00 01 F4C 01743 00610 F 00118300LRU 731BC          AMP
C155  6685001155E0EBF 2 RR C0 C1 F4C CC790 C0656 G 00147100LRL 41134          TMF CTL
0156  1270000238954    2 RRRC3 01 F4C C1623 CC763 L CC15E4CCLRL 73S1C          CTL CURS
C157  1270000233663    2 RRRC3 01 F4C 0E346 01396 L 0415EECCLRL 73S3C          COMP BAL
C158  12700C5518452    1 RRRC3 01 F4D CC5C0 C0075 L C4156600LRU 73570          COMP
C159  1430000541C65BF 2 RRR03 01 F4C C3783 C018S G CC3EEECCLRL 74370          DFV TNG
C160  1430CC13CC516EF 1 RRR03 01 F4C 02203 00221 G 0C12C0C0LRU 747JC          RDR CTL
C161  1430OC141S504BF 2 RRRC3 01 F4D C37EC C041E G C13CCCCCLRL 74CC0          FSC IND
C162  1430001669989EF 2 RRR03 01 F4C 02569 00283 G 013CCCCCLRL 74DBC          PIL IND
C 63  1430CC172753SBF 2 RRR03 01 F4D 35520 01421 G 05188600LRL 74710          SYNCH
```

```
      0000000001111111111122222222223333333333444444444455555555556666666666777777777
      1234567890123456789012345678901234567890123456789012345678901234567890123456789
CARD
0164  1430CC2215EE0EBF 2 RRRC3  01 F4D 14760 00738 G C35CC0CCLRL 74EAC        ICL
0165  1430CC41CCE58BF  2 RRRC3  C1 F4D 18485 01254 G C6C4E5CCLRL 74730        ANT
0166  1430CC43E2362BF  2 RRR03  01 F4D 19733 01776 G C4C1C2CCLRL 7479C        RCVR TRN
0167  1430CC48CC211BF  2 RRRC3  01 F4D 06338 00063 G 00555100LRU 74770        ANT CTL
0168  1430008114718BF  2 RRR03  01 F4D CE426 CC26E G CC22CCCCLRL 74A10        ACC
0169  1430009183652BF  1 RRR03  01 F4D 02454 C0C58 G CC5352CCLRL 74760        FIL IND
0170  1430CC51E3E53EF  1 RRRC3  01 F4C C4173 C0125 G 004E6700LRU 74750        FSC IND
0171  1430009183654BF  2 RRRC3  01 F4C 23567 C115E G C455E1CCLRL 74720        CTLFWRSP
0172  1430CC53C805EEF  1 RRRC3  01 F4C 00531 00266 G 025C110CLRU 747EC        ICL
0173  66050083E3333    2 RRRC3  01 F4D 10001 01400 H 01372100LRU 71E20        AMP
0174  1430010121587EF  2 RRR03  01 F4C C5543 C0412 G C1152CCCLRL 74A2C        CCMF
0175  66050C83E3338    2 RRRC3  01 F4D 12084 05800 H 0321E2C0LRL 71E1C        COMP CTL
0176  1430CC51555678F  2 RRRC2  C1 F4D CCE57 CCC2E G CCC5EEECLRL 747AC        CTL
0177  1430CC13C0515BF  2 RRR02  01 F4C 02058 00124 G CC12CCCCLRL 74740        RDR CTL
0178  1270CCC23E562    2 RRRC2  01 F4C 02915 01367 L C0449600LRL 73520        CTL COMP
0179  66050094578 35   2 RRR04  01 F4D C5192 C0363 H C212E5CCLRL 71H50        CSCU
018C  66050C9940194     2 RRR04  01 F4C 19358 C0194 H 04CECECCLRL 71E20        COMP
0181  1270600231042     2 RR CO  C1 F4C CC574 C0515 L C0171500SRU 7353T        SRVO
0182  1270003461979     1 RR CO  01 F4D C0313 CCC47 L C13245CCSRL 73CLC        INT ELAY
0183  1270C3455E7C      1 RR CO  01 F4C 00713 00535 L C1326C0CSRU 73CGC        GMBL
0184  1270003E52210     1 RR CO  C2 F4D C0300 0030C L 00363570SRL 730GA        GYRC
0185  1270003938955     1 RR 00  01 F4D C0043 C00C7 L CCE7E1CCSRL 73C1C        FCC CTL
0186  1270CC47E2473     2 RR 00  01 F4D 00289 00272 L 0CE3CCCCSRL 7354X        CHAS AY
0187  1270CC5E1E449     1 RR CO  C1 F4D CC313 CC2E1 L CC73CCCCSRL 7357A        CHAS
0188  1270CC51EC176     2 RR 00  01 F4C 00357 C0325 L CC2162CCSRL 7357P        SRVC
0185  1270CC51EC178     2 RR CO  01 F4C CC500 C044C L C025E400SRL 7357M        FTCH SRV
0190  1270009755811     2 RR 00  01 F4D C0229 CC137 L CC21CCC0SRL 7353B        FWR SUP
C191  1270CC9755822     2 RR 00  01 F4C CC569 C0E34 L 0C2E5CCCSRL 7353K        SRVC BGT
C192  1270CCC575E5E5    2 RR CO  C1 F4D CC356 C036C L C0125000SRU 7353H        SRVC
0193  1430000139123BF  2 RR CO  01 F4D C2124 CC27E G CC25CCCCSRL 74AEC        LEAD ANG
C194  1430CC013512EEF  2 RR 00  01 F4C 02691 00404 G 0022CC00SRU 74A3C        ALL STRG
C195  1430CC0E5ECCEBF  2 RR CO  C1 F4D CC568 C0062 G 0C0835305FU 7478T        IND
C196  1430000784192EF  2 RR 00  01 F4D 0C643 CC05E G CCC75ECCSRL 747EJ        IND CTL
C197  1430C01CEE78EEF  2 RR CO  01 F4C 00400 00231 G 0041ECCCSRL 7413H        MXR AY
C198  1430CC1075E4BF   2 RR CO  C1 F4D CC108 C0065 G C0CE515CSRL 74CEH        VLT REG
C199  1430CC13415C5EF  2 RR 00  02 F4C 00940 00940 G 00124E0CSRU 7414G        RATE GYR
C200  1430001355E3CBF  2 RR CO  C1 F4D C7591 00226 G 0C4CC000SRU 7511A        CTL CHAS
0201  1430002558597EF  2 RR 00  01 F4D CC507 C0340 G CCC8CCCCSRL 74DCG        PWR SUP
C202  1430CC3033E27EF  2 RR 00  02 F4D 00806 00733 G 0CCE4CCCSRL 74DCJ        DEF AMP
0203  1430003051545BF  2 RR CO  C1 F4D C1E63 01114 G CC153ECCSRL 74DA2        DEF AMP
C204  1430CC41CC852EF  2 RR 00  01 F4C 02649 02331 G CCCE3CECSRL 7473H        MRG HARN
C205  1430CC4835E518F  2 RR CO  01 F4C CC760 00659 G 0004155CSRU 74150        REFLECT
0206  1430007732913BF  2 RR 00  01 F4D CC715 CC715 G CCC5475CSRL 7411B        AMP
C207  1430CC7732925EF  2 RR 00  01 F4C C1623 00308 G 00C574CCSRL 7412C        GEN
C208  1430CC7751142BF  2 RR CO  02 F4D C1228 C1117 G 0011740CSRU 7414J        SERVO AY
0209  1430007751143EF  2 RR 00  01 F4C C2385 02C7E G CC1123CCSRL 7413E        AMP
C210  1430CC7754185EF  2 RR 00  01 F4C 00460 00069 G 000E263CSRU 7479D        MTR BX
0211  1430CC7754195BF  2 RR CO  C1 F4D 03412 02661 G 0C16740CSRL 7479G        AMP
0212  1430007754191EF  2 RR 00  01 F4D CC643 00674 G CCC434CCSRL 7475N        AMP
C213  1430CC7754192EF  2 RR CO  01 F4C 00317 00295 G C0CE6E5CSRU 74130        CW FILTR
0214  1430CC6023C56BF  2 RR CO  C1 F4D CC603 CC711 G CCC15CECSRL 7473J        RCT JNT
C215  1430CC8C5C771EF  2 RR 00  01 F4C 00652 00605 G CCC345CCSRL 7414M        MAG STF
C216  1430CC8223C97BF  2 RR 00  01 F4C CC500 00455 G C0C7725CSRU 7473E        RCT JNT
0217  1430008578936BF  2 RR 00  01 F4D 01034 CCC72 G CC245CCCSRL 74A50        HEAD AIM
C21P  1430C0915C0035EF 2 RR CO  01 F4C C2474 C2280 G CC3E46CCSRL 7471J        MOD TIME
```

```
          0000000001111111111222222222233333333334444444445555555555666666666677777777778
          1234567890123456789012345678901234567890123456789012345678901234567890123456789
CARD
C219      1430CC91S0036EF  2 RR CO 01 F4D CC475 C0432 C CC371CCCSRL 7476C          ACT
C220      1430CCS1SCC37BF  2 RR CO C1 F4D 01496 01376 G 00165400SRU 7414C          ACT AZ
0221      1430CC91SOC61BF  2 RR CO C1 F4D C2342 C1CC7 G CC16S3CCSRL 7472E          FLL CLMB
C222      1430CCS24P88SEF  2 RR 00 01 F4C 01781 C1710 G 0025E3CCSRU 74738          BAL AY
C223      1430CC93115948F  2 RR CO C2 F4C CCC17 00008 G C005S61CSRU 747GA          LG VCLUT
0224      1430C093115S7EF  1 RR CO 02 F4C C0C34 C0013 G CCCE8SCCSRL 747HA          CKT CD
C225      1430CC94416E7EF  2 RR CO 01 F4C 04907 04514 G C0C026CCSRL 7471F          TIME DIS
C226      1430CCS4CC165BF  2 RR CO 01 F4D CCE82 00621 G 0C19720OSRU 7471N          IF AMP
C227      1430010121563EF  2 RR 00 01 F4C C5002 C0412 G CC32SCCCSRL 74A40          RNG INT
C228      163CC1C182115    2 RR CO 01 F4D 01088 01088 G 00CS715CSRU 1343F          CKT CD
C229      1630010216543    2 RR CO C1 F4D C1CEE CCE7C G CC22CECCSRL 1343G          CTL EX
C23C      1630C10345997    2 RR 00 01 F4C 01C88 010BE G CCC2176CSRL 13438          CKT CD
C231      66C5CCCS1C97E    2 RR CO 01 F4C 01000 00E71 F C0205200SRU 71E2E          AMF
C232      66050005S3863    2 RR CO 01 F4D CCE29 00S75 F CCC2S2SCSRL 71H2L          AMF
C233      66C5CCCS53866    2 RR 00 01 F4C C1S50 C1E53 F CCC4S030SRL 71F2Q          AMF
C234      66C5CCCS55E6E    2 RR CO C1 F4C CC900 C0837 F 00032140SRU 71F3G          AMF
0235      66050005S5371    2 RR 00 C1 F4D C1543 C1E47 F CCC327SCSRL 71F3L          REG
C236      66C5C00563113    2 RR CO 01 F4C 01100 0105E F 0CC73440SRL 71H2U          AMP
0237      66C5CCC5S63115   2 RR CO C1 F4D C1475 C130C F 00C32240SRU 71F3J          PWR SUP
0238      66050005E3118    2 RR CO C1 F4C 01428 01314 F CC174CCCSRL 71H4A          CTL
C239      66C5CCCS5C3115   2 RR CO 01 F4C 01501 01441 F 0CCE54SCSRU 71F4F          GVEN
0240      66C5CC05E43123   2 RR CO C1 F4D C7581 C72C2 F CCE31CCCSRL 71F4R          INT SHFT
C241      66C5CC1C95425    2 RR 00 01 F4C CC500 C0846 F CCC34E1CSRL 71H2N          AMF
C242      66C5CC1C63C7S    2 RR CO 01 F4D 01063 00991 F 00C38110SRL 71F2S          GCNOGO
C243      66C5CC72E3497    2 RR CO C1 F4D C2555 C2725 F CC33C33CCSRL 71F48          PWR SUF
C244      66C5CC83E5407    2 RR 00 01 F4C 01332 C1187 F CCC473ECSRL 71H3A          LVL DET
C245      66C5CCE5S29CE    2 RR CO C1 F4C CC769 C0746 F 00C92240SRL 71E2E          AMF
C246      66C5CC8C57C96    2 RR CO 04 F4C CCE25 CC554 F CCC65C7CSRL 71E2A          AMF
C247      66C5CC8C97C97    2 RR CO 01 F4C 00710 00E47 F CC1639C0SRL 71B2D          AMP
C248      66C5CC92E5120    2 RR CO 0E F4C CC227 00211 F 00C37290SRU 71F3N          AMF
C249      6605C094S0273    2 RR CO 1S F4C CC243 C0236 F CCCE53CCSRL 71H3L          AMF
C25C      66C5CC5E42777    2 RR 00 01 F4C 01134 01100 F 00311400SRU 71F5A          SRVC
0251      66C5CC5E45746    2 RR CO C1 F4D CCE33 C0E0C F 00C73E7CSRU 71E2C          AMF MOD
0252      66C5CC96E5749    2 RR 00 C1 F4C C0470 C0432 F CC14E4CCSRL 71E2F          AMF
C253      66C5CC594C15E    2 RR CO 01 F4D 05064 04760 F 007E54CCSRL 71E2A          CCMP SHL
C254      1630CC27E5E4S    2 RR CO 01 F4D C1E26 CC351 G CCC7543CSRL 1344J          FCUSNG
C255      1C05CCC178806    2 RR 00 24 F4C CCC00 C0000 L CCCCE437E0Q 75D1E          BCLT AY
C256      1C05CC2213054    2 RR CO C1 F4D CCC00 C0000 L CCC2S704ECG 7546G          DRIVE AY
C257      1005C0C9E493E    2 RR CO 03 F4C CCCCC C0000 L C14SC1CCNOF 75C00          GAU 4-A
C258      1C05CC8521898    2 RR 00 C4 F4C C0000 00000 L CCCES3CCNOP 7546C          FEED ASY
C259      1C05CCES53722    2 RR CO C2 F4C CCCC0 C0000 L 00117700NCF 7547A          HOUSING
0260      1005CC90E0800    2 RR CO C1 F4C CCCCC C0CC0 L CCC1EEC2ECG 7546R          CHUTE
C261      1C05CCSC71838    2 RR CO 01 F4C 00000 00000 L 00C1641BECG 7540C          CHUTE
C262      1005CCS017514    2 RR CO C2 F4D CCCC0 C0000 L 00C32E2CNCF 75C99          DISC
C263      1005CC90B3824    2 RR 00 04 F4C CCC00 C0000 L CC114SCCNCF 7546E          INNER DM
C264      1C05CCSCE3E25    2 RR CO 02 F4C 00000 00000 L 00CS43CCNCF 7546K          EXITUNIT
C265      1C0ECCS1EC28E    2 RR 00 01 F4D CCCCC C0C0C L CCCE5410NCF 75C1C          FAIRING
C266      1005CC94E8405    2 RR 00 02 F4C CCC00 C0000 L CCCSCEE3EOG 75D1F          BRAKE
C267      1C05CCS4E8414    2 RR CO 04 F4C CCC00 C0000 L CC25E40CNCF 75C1M          GUNSTART
C268      1005C09E43297    2 RR CO C1 F4C CCCCC CCCCC L CCCE3173EOG 75C1E          EJECTCR
C269      1C95C0025E657    2 RR 00 01 F4C CCCC0 CCCCC L CC1E21CCNCP 75XXX          MAL 12CA
C27C      1C95CCES97E12    2 RR CO 02 F4C CCCC0 C0000 L 0035100CNCF 75C1J          PANEL
C271      1025J09842060    2 RR CU 13 F4C CCC00 C0000 L CCCC2718ECG 75135          CAF AY
0272      1C95CC99EC05E    2 RR CO 01 F4C C0000 00000 L 005204CCNCP 75IRG          AERO 27A
C27       1430CC02CE41F    2 RR CO C1 F4D CC000 C0000 G CCCC7E70ECC 74319          AMFLIFIE
```

```
      CC00CCCC(1111111111222222222233333333334444444444555555555566666666667777777778
      1234567890123456789012345678901234567890123456789012345678901234567890123456789
CARD
C274  14300C0E51C52   2 RR CO C1 F4C CCC00 C0000C C CCC4E815ECC 7418E          INDICAT
0275  1430007729378   2 RR 00 01 F4D CCC00 CCCCC G CCC157C4E0C 7472K          FWRSLEAY
C276  1430C0E1E52E4EF 2 RR CO C8 F4C 00000 00000 C CCCC2176ECC 74EV0          FILTER
0277  14300C6E145278F 2 RR CO C1 F4D CCC00 CCCCC G 00C3E300ECC 74E0C          CIRC CC
C278  1440000513187AB 2 RR 00 01 F4C 00000 C0CCC L CCC52EECNOP 7515D          FCN SUF
C279  144CCC1043473EL 2 RR 00 01 F4C 00000 00000 L CCE0CC00NCP 7513C          AERC 7A
0280  14400C1043474BL 2 RR CO 01 F4C CCCCC C0CCC L CCECCCCCNCF 7513D          AERC 7A
C281  1440001360807AE 2 RR 00 01 F4C CC000 C0C0C L CCCEC770NCP 7E11B          FCN SUF
0282  1440CCE745675AB 2 RR 00 01 F4C C0C0C C0CC0 L 0012S400NOP 7E110          LAUNCHER
0283  14400096E52220   2 RR CO C1 F4D CCCC0 CCCCC L CCCC7155ECC 7515C          MECHANIS
C284  1680CCCC73554     2 RR 00 01 F4C CC000 C0CC0 G 00C25E47EOC CCC0C          GRIP AY
C285  1560CCE7C7917     2 RR CO C1 F4C CCC00 C0000 C 00031544ECC 000FE          PRCEE FE
0286  1560006872027     2 RR 00 02 F4D CCCCC CCCCC G CCCC2E45ECC 4E1CC          SWIVELAY
C287  1630C022E2374       RR CO 96 F4C 00000 00000 C CCCCE351ECC 13400          DISC
C288  1630CC22E2375     2 RR CO 84 F4D CCC00 C0CCC G CCC02058ECC 13400          DISC ST
0289  16300022E2376     2 RR 00 10 F4C CC000 C0CCC G CCC13161EOC 134CC          PLATE
C290  1630CC4413778     2 RR 00 21 F4C C0000 00000 F 00165ECCNCF 132BA          WHEEL MN
C291  1630CC49E3225     2 RR CO 1C F4C CCCC0 CCCCC G CCC11449ECC 13400          PLATE
C292  1630CC8521432     2 RR 00 12 F4C CCC00 C0CCC C CCC251EEECC 133CC          WHEEL NC
C293  1650CCCCE3E95     2 RR CO 06 F4C 00000 00000 F 01133000NCF 2273C          SPEED DR
C294  16500007C5241     2 RR CO 01 F4D CCCC0 C0CCC G CCCC175E4ECC 4513Y          SWIVEL
C295  1650CCC841210     2 RR 00 02 F4C C0000 C0000 G CCCC225EEOC 4521H          SWIVEL
C296  1650C044E1345     2 RR CO C1 F4C CC000 C0000 C CCCC185SECC 1331J          SWIVEL
0297  16500066128A0     2 RR 00 02 F4D CCCCC C0CCC H CCCC1E75ECC CCC1K          FILTER
C298  1660CC42CE437     2 RR CO 03 F4C 00000 00000 C CCCC4232ECC 71LXC          CABLE
C299  1670C052SE712     2 RR CO 02 F4D CCCCC C0CCC F CCCC175S1ECC CCCCC          CHLIE
C300  167CCCE333311     2 RR CO 02 F4C 00000 00000 F CCC174E2ECC S321J          CANOPY
C301  26200C0E84523     2 RR CO E3 F4D CCCCC C0C0C G 0001EECSNCF 13300          TIRE
0302  2640002224525     2 RR 00 04 F4C C0C00 C0C0C K CCCCC1CCEOC CCCCC          CAP
C303  2640CC81C5EE1     2 RR 00 04 F4C 00000 00000 K 00CCC1C0ECC 00000          CORE
0304  2835006825352     2 RR 00 01 F4D CCCCC C0CCC F CC1C51CCNCF AAEB4          VALVE
C305  2835008619955     2 RR 00 01 F4C CC000 C0000 P CCCC71C2EOC AG1EA          REG PRE
C306  2840005715E5E     2 RR CO 02 F4C CCCC0 C0C0C P 00C1E16EEOC 23500          FLAP
0307  2620000824222     2 RR CO 75 F4D CCC0C C00CC G CCC2S434ECC 13300          TIRENCSE
C308  2E400C953CE74     2 RR 00 09 F4C 00000 C0000 P 0CC0S126EOC CCCS3          BLADE
C309  2840CC9S5E4E3     2 RR CO C1 F4C CCC00 C0000 P C0C2CC73ECC 23315          LINER
C310  2840005S5E4E4     2 RR CO 02 F4D CCC00 C00CC P CCCCEC7EECC 23317          LINER
C311  2840CC5S5E4E5     2 RR CO 01 F4C 00000 00000 P C0C2C25SECC 23316          LINER
C312  2S10CC49254E3EYP  2 RR CO C3 F4D CCCC0 CC000 F 00224000NCF AAEHL          CONTROL
0313  2910006470721     2 RR 00 03 F4C C0C00 C0000 S CCCC2SS4EOC CCCC0          CABELRET
C314  2S10CCS1CE45SYP  2 RR 00 03 F4C 00000 00000 F 0014S1CCNCF AAEHL          CONTROL
0315  2920006ECCCSEEYP  2 RR 00 01 F4D CCCCC CC0CC F CCC7E22CNCF AACFC          STARTER
C316  2920006024259     2 RR 00 01 F4C CCC00 CC0C0 S CCCC7ECCECC AG1SS          MAGNETC
C317  2920CE47C71E      2 RR 00 01 F4C CCC00 CC0CC S C0C10088ECC AG153          GEN STRT
0318  2920006SE7336     2 RR CO C1 F4D CCC00 CCC0C S CCCC4223EECC AG100          MAG
C319  2920CC88C3428     2 RR 00 01 F4C C0000 C0C0C S C0CC73E4E0C AG1SS          MAG
C32C  2920CCS77EE31     2 RR CO C1 F4C C0C00 C0C0C S CCCC7176ECC AG1SS          MAGNETO
0321  2945008548795     2 RR CO 02 F4D CCCC0 CCCCC H CCCC725SEEC 2371R          ELEMENT
C322  2S45CC875E021     2 RR CO 01 F4C 00000 00000 P 0CC133033ECC 2371S          ELEMENT
0323  4320CC7E54539YZ   2 RR CO 01 F4D CCC0C C0CCC F 00C19420NCF AE172          PUMP
0324  473000812304C     2 RR 00 02 F4C C0C00 C0C0C S CCC133312E0C AAEHE          COUPLING
C325  4E2CCCC34E203     2 RR CO 01 F4C 00000 00000 C 00C1C275ECC 1230C          VALVE AY
0326  4E20CCE0E1E22     2 RR CO 02 F4D CCCCC CCCCC S CCC23ECEECC CCC00          VALVE
C327  5310CC0113090     2 RR 00 01 F4C CC000 C0C0C S CCCCC1CCEOC CCCCC          FELT
C328  5310CCC0113C51    2 RR CO 01 F4C 00000 00000 S C0000100ECC 00000          FELT
```

```
      CCCCCCCCC111111111122222222223333333333444444444455555555566666666667777777778
      123456789C123456789C123456789C123456789C123456789C123456789C123456789C123456789C
CARD
0329  5325CCC0S2CS37    2 RR CO 01 F4C CCCOO COOCC S CCCCC100ECC 00000          CRCNMET
0330  5330C04705S40    2 RR 00 01 F4C CCCOO COOOO S CCCCC1CCEGG CCCGC          SEAL
C331  5330CCSS52735    2 FF CO 01 F4C 00000 COOOC S COCCC1CCECC CCCCC          PACKING
0332  5330000E1E4950    2 RR CO 03 F4D COCOO 00000 S COCOC1COECC C0000          PACKING
0333  5330CC8415777    2 RR 00 01 F4C CCCOO CGCOC S CCCCC1CCEOC CCCCC          PACKING
C334  5330CCS4EE22S    2 RR CO 01 F4D 00000 00000 S 00CCC1CCECC 00000          PACKING
C335  SE41CCS7J6185    2 RR CO C1 F4D CCCCC CCCCC 3 CCCC3ECCECC 71LGC          CTL
C336  5S350C6BE0447    2 RR 00 02 F4C 00000 COCCC S CCCC2C44EOC CCC00          CCNNECIC
C337  55ECCCE42S578    2 RR CO 01 F4C COCOO COOOC C C02447EEECG 747GC          DVS TUEE
C338  5960009311572    2 RR 00 01 F4D CCCCO CCCCC G CC1223E2EOC 747GC          DVS TUEE
C33S  6105CC97337247    2 RF 00 01 F4C CCCOO COOOO F OCC1S7S3ECC 41140          MOTCR
C340  61100C727C792    2 RR CO 01 F4C CCCOO COOCO F 00176400NCF AANCH          REG
0341  6110008772470    2 RR 00 01 F4C CCCCC COCOC F CCC1715CNCF AAEEC          REG
C342  6110CC92SSSS4    2 RR CO 01 F4C C0000 00000 F 0016E5S0NCF AAEAK          CCNTROL
0343  611500CES1405    2 RR CO C1 F4D CCCOO COOOC F C03342CONCF AAEC1          GEN
0344  61150C9031256EF  2 RR 00 06 F4C CCCOO COCOC G CC174ECCNOF 42210          GEN
C345  6140CC57E7525    2 RR 00 02 F4D 00000 00000 F CCCEESSONCF AAEEA          EATTERY
C346  6140CC7EE1725    2 RR CO C4 F4D CCCCO CCCCC F CCC2S34CECG 42330          EATTERY
C347  6605006518187    2 FF 00 02 F4C 00000 COOOC F CCCCS34EEOO S12AB          CCMPASS
C34E  668CCCSC75703    2 RR 00 01 F4C CCCOO COOOC C 00C122C9ECC 4E42K          TANKUNIT
C349  6680009114290    2 RR CO 01 F4D CCCCC CCCCC G CCCCS3S1ECG 46219          TANKUNIT
C35C  6680009114291    2 FF 00 01 F4C C0000 C0000 C COC10E40ECC 4E1AS          TANKUNIT
C351  668000S114295    2 RR CO 01 F4C CCCOO CCOOC C 00C11845ECC 4E42N          TANKUNIT
C352  6680009114297    2 RR CO 01 F4C CCCCC CCOOC G CCC1CEE2ECG 4E42F          TANKUNIT
C353  668CCCC9114298    2 RR 00 01 F4C 00000 00000 C 00C12C91ECC 4E42L          TANKUNIT
0354  6680005142702    2 RR CO C1 F4D CCCOO C0000 G CCC1C277ECC 46100          TANKUNIT
0355  6680009451205    2 RR 00 01 F4C CCCOO C0000 G CCC2765ZEOC 4E4CC          TANKUNIT
C35E  6EEBCCC54E555E    2 RR CO 01 F4C 00000 00000 G COC124CCECG 4E42C          TANKUNIT
0357  6680CC9531815    2 RR CO 01 F4D CCCOC C0000 G CCC17115ECC 46100          TANKUNIT
C358  $ENDLIST
```

APPENDIX D.2

LISTING OF

PHASE 1 AND PHASE 2 F4D INPUT DATA

Phase 1 Data

1	1	25927.00	0.65423	RR	LRU
2	1	3054.00	0.01932	RR	LRU
3	1	600.00	0.03737	RR	LRU
4	1	8166.00	0.04807	RR	LRU
5	1	11107.00	0.02875	RR	LRU
6	1	3162.00	0.03737	RR	LRU
7	1	13900.00	0.01276	RR	LRU
8	1	2332.00	0.04025	RR	LRU
9	1	2868.00	0.03599	RR	LRU
10	1	3550.00	0.03599	RR	LRU
11	1	8784.00	0.07348	RR	LRU
12	1	12366.00	0.04025	RR	LRU
13	1	1627.00	0.25553	RR	LRU
14	1	2369.00	0.16422	RR	LRU
15	1	5258.00	0.11396	RR	LRU
16	1	1325.00	0.03231	RR	LRU
17	1	2975.00	0.12086	RR	LRU
18	1	2520.00	0.10879	RR	LRU
19	1	303.24	0.13075	RR	LRU
20	1	799.00	0.05048	RR	LRU
21	1	1424.00	0.06348	RR	LRU
22	1	1056.00	0.05773	RR	LRU
23	1	845.00	0.04025	RR	LRU
24	1	919.80	0.04726	RR	LRU
25	1	717.70	0.03749	RR	LRU
26	1	927.00	0.09522	RR	LRU
27	1	3126.00	0.10223	RR	LRU
28	1	2122.00	0.06049	RR	LRU
29	1	6891.00	0.21332	RR	LRU
30	1	582.00	0.11569	RR	LRU
31	2	3745.00	0.10948	RR	LRU
32	1	692.70	0.05198	RR	LRU
33	1	1333.00	0.07383	RR	LRU
34	1	1333.00	0.08889	RR	LRU
35	1	3202.00	0.22850	RR	LRU
36	1	516.00	0.71875	RR	LRU
37	1	1241.00	0.29003	RR	LRU
38	1	2105.00	0.05750	RR	LRU
39	1	824.50	0.07820	RR	LRU
40	1	1342.00	0.06014	RR	LRU
41	2	3914.00	0.17480	RR	LRU
42	2	891.73	0.17227	RR	LRU
43	2	933.80	0.04692	RR	LRU
44	2	2058.00	0.19343	RR	LRU
45	2	349.10	0.04853	RR	LRU
46	2	1242.00	0.06923	RR	LRU
47	2	471.70	0.38180	RR	LRU
48	5	304.30	0.27370	RR	LRU
49	1	1317.00	0.04393	RR	LRU
50	2	2083.00	0.05865	RR	LRU
51	2	1044.00	0.15502	RR	LRU
52	2	637.90	0.13156	RR	LRU
53	2	487.90	0.38318	RR	LRU
54	2	9901.00	0.32798	RR	LRU
55	2	275.30	0.06624	RR	LRU
56	1	2975.00	0.17043	RR	LRU
57	4	1130.00	0.96370	RR	LRU
58	1	886.00	0.07624	RR	LRU
59	5	320.30	0.14087	RR	LRU
60	1	1300.00	0.04174	RR	LRU
61	1	1300.00	0.05566	RR	LRU
62	1	456.90	0.06175	RR	LRU
63	1	309.70	0.04335	RR	LRU
64	1	945.50	0.20493	RR	LRU
65	1	288.00	0.12121	RR	LRU

66	2	978.00	1.10078	RR	LRU
67	2	708.00	0.72450	RR	LRU
68	1	4000.00	0.20412	RR	LRU
69	2	412.00	0.01035	RR	LRU
70	2	262.10	0.06992	RR	LRU
71	2	1358.00	0.84111	RR	LRU
72	1	1874.00	0.17284	RR	LRU
73	1	280.00	0.16422	RR	LRU
74	1	3658.00	0.06382	RR	LRU
75	1	335.00	0.09579	RR	LRU
76	2	2129.00	0.37237	RR	LRU
77	1	2166.00	0.14973	RR	LRU
78	1	1158.00	0.12788	RR	LRU
79	1	3550.00	0.19251	RR	LRU
80	1	3290.00	0.43493	RR	LRU
81	1	902.40	0.04715	RR	LRU
82	1	1115.00	0.25392	RR	LRU
83	2	267.80	0.23000	RR	LRU
84	1	1051.00	0.54383	RR	LRU
85	1	875.50	0.13524	RR	LRU
86	2	786.50	0.24978	RR	LRU
87	1	1401.00	0.14846	RR	LRU
88	1	1585.00	0.06290	RR	LRU
89	1	6180.00	0.84628	RR	LRU
90	1	2553.00	0.05186	RR	LRU
91	2	2292.00	0.11707	RR	LRU
92	1	831.20	0.05681	RR	LRU
93	1	641.69	0.11638	RR	LRU
94	1	1421.00	0.11500	RR	LRU
95	1	1656.00	0.19090	RR	LRU
96	1	165.00	0.13570	RR	LRU
97	1	1656.00	0.07003	RR	LRU
98	1	591.20	0.06359	RR	LRU
99	1	21858.00	0.66447	RR	LRU
100	2	1275.00	0.12144	RR	LRU
101	1	2086.00	0.08820	RR	LRU
102	1	253.60	0.03772	RR	LRU
103	1	605.64	0.48567	RR	LRU
104	1	1308.00	0.21689	RR	LRU
105	2	457.30	0.41814	RR	LRU
106	1	1224.00	0.27174	RR	LRU
107	1	1359.00	0.10718	RR	LRU
108	1	4000.00	2.44950	RRR	LRU
109	1	2560.00	0.64124	RRR	LRU
110	1	12389.00	0.27715	RR	LRU
111	2	430.50	0.08027	RR	LRU
112	1	1082.00	0.06382	RR	LRU
113	1	892.00	0.07682	RR	LRU
114	2	1429.00	0.06003	RR	LRU
115	2	1400.00	0.16882	RR	LRU
116	2	250.00	0.06118	RR	LRU
117	1	471.70	0.10154	RR	LRU
118	1	719.70	0.51589	RR	LRU
119	1	1686.00	0.08096	RR	LRU
120	1	1449.00	0.25001	RR	LRU
121	2	970.50	0.09177	RR	LRU
122	1	1278.00	0.12776	RR	LRU
123	1	886.00	0.11178	RR	LRU
124	3	659.40	0.08901	RR	LRU
125	1	1119.00	0.08636	RR	LRU
126	2	702.60	0.48323	RR	LRU
127	1	5954.00	1.02649	RR	LRU
128	1	2380.00	0.52808	RR	LRU
129	1	650.00	0.09016	RR	LRU
130	1	480.00	0.01035	RR	LRU
131	1	840.00	0.07371	RR	LRU

132	1	5000.00	0.62732	RR	LRU
133	1	1010.00	0.10718	RR	LRU
134	1	718.75	0.10430	RR	LRU
135	2	279.00	0.26818	RR	LRU
136	1	129.00	0.07728	RR	LRU
137	1	3097.00	0.71875	RR	LRU
138	2	696.00	0.20332	RR	LRU
139	1	3971.00	0.70081	RR	LRU
140	1	2760.00	0.48288	RR	LRU
141	1	2711.00	1.04075	RR	LRU
142	1	9674.00	1.46027	RR	LRU
143	1	3275.00	0.08119	RR	LRU
144	2	1615.00	0.87929	RR	LRU
145	1	236.10	0.03875	RR	LRU
146	1	1674.00	0.04611	RR	LRU
147	2	436.00	0.20102	RR	LRU
148	1	35511.00	1.19232	RR	LRU
149	1	1472.00	0.12960	RR	LRU
150	1	2730.00	0.20838	RR	LRU
151	1	17694.00	0.99969	RR	LRU
152	1	3914.00	0.34258	RR	LRU
153	1	3223.00	0.20642	RR	LRU
154	1	1183.00	0.20044	RR	LRU
155	1	1471.00	0.09085	RR	LRU
156	1	1984.00	0.74658	RRR	LRU
157	1	41566.00	2.91916	RRR	LRU
158	1	41566.00	0.23000	RRR	LRU
159	1	3668.00	1.74018	RRR	LRU
160	1	1200.00	1.01338	RRR	LRU
161	1	13000.00	1.73880	RRR	LRU
162	1	13000.00	1.18174	RRR	LRU
163	1	51886.00	16.33919	RRR	LRU
164	1	39000.00	6.78960	RRR	LRU
165	1	60485.00	8.50310	RRR	LRU
166	1	40102.00	9.07718	RRR	LRU
167	1	5951.00	2.91548	RRR	LRU
168	1	2200.00	3.87596	RRR	LRU
169	1	5352.00	1.12884	RRR	LRU
170	1	4687.00	1.91958	RRR	LRU
171	1	45581.00	11.02481	RRR	LRU
172	1	29011.00	0.24426	RRR	LRU
173	1	13721.00	4.60046	RRR	LRU
174	1	11520.00	4.38978	RRR	LRU
175	1	32182.00	5.55864	RRR	LRU
176	1	566.50	0.29566	RRR	LRU
177	1	1200.00	0.71001	RRR	LRU
178	1	4496.00	1.00567	RRR	LRU
179	1	21265.00	2.98540	RRR	LRU
180	1	40606.00	11.13084	RRR	LRU

Phase 2 Data

1	0	1	14.38937	25927.00	RR	LRU
2	0	1	0.33633	3054.00	RR	LRU
3	0	1	0.65065	600.00	RR	LRU
4	0	1	0.83683	8166.00	RR	LRU
5	0	1	0.50050	11107.00	RR	LRU
6	0	1	0.65065	3162.00	RR	LRU
7	0	1	0.22222	13900.00	RR	LRU
8	0	1	0.70070	2332.00	RR	LRU
9	0	1	0.62662	2868.00	RR	LRU
10	0	1	0.62662	3550.00	RR	LRU
11	0	1	1.27928	8784.00	RR	LRU
12	0	1	0.70070	12366.00	RR	LRU
13	1	1	4.44843	1627.00	RR	LRU
14	0	1	2.65685	2369.00	RR	LRU
15	0	1	1.98398	5258.00	RR	LRU
16	0	1	0.56256	1325.00	RR	LRU
17	0	1	2.10409	2575.00	RR	LRU
18	0	1	1.89388	2520.00	RR	LRU
19	1	1	2.27625	303.24	RR	LRU
20	0	1	0.87888	799.00	RR	LRU
21	0	1	1.10510	1424.00	RR	LRU
22	0	1	1.00500	1056.00	RR	LRU
23	0	1	0.70070	645.00	RR	LRU
24	0	1	0.82282	919.80	RR	LRU
25	0	1	0.65265	717.70	RR	LRU
26	1	1	1.65765	927.00	RR	LRU
27	0	1	1.77977	3126.00	RR	LRU
28	0	1	1.05305	2122.00	RR	LRU
29	0	1	3.71369	6891.30	RR	LRU
30	1	1	2.01401	582.00	RR	LRU
31	0	2	1.90589	3745.00	RR	LRU
32	0	1	0.90490	692.70	RR	LRU
33	0	1	1.28828	1333.00	RR	LRU
34	0	1	1.54754	1333.00	RR	LRU
35	0	1	3.97797	3202.00	RR	LRU
36	3	1	12.51249	516.00	RR	LRU
37	1	1	5.04903	1241.00	RR	LRU
38	0	1	1.00100	2105.00	RR	LRU
39	1	1	1.36135	824.50	RR	LRU
40	0	1	1.04704	1342.00	RR	LRU
41	0	2	3.04303	3914.00	RR	LRU
42	1	2	2.99898	891.73	RR	LRU
43	0	2	0.81682	933.80	RR	LRU
44	0	2	3.30735	2058.00	RR	LRU
45	1	2	0.84484	349.10	RR	LRU
46	0	2	1.20520	1242.00	RR	LRU
47	2	20	6.64663	471.70	RR	LRU
48	2	5	4.76474	304.30	RR	LRU
49	0	1	0.76476	1317.00	RR	LRU
50	0	2	1.02102	2083.00	RR	LRU
51	1	2	2.69867	1044.00	RR	LRU
52	1	8	2.29027	637.50	RR	LRU
53	2	2	6.67065	487.90	RR	LRU
54	0	2	5.70970	9901.00	RR	LRU
55	1	2	1.15315	275.30	RR	LRU
56	0	1	2.96695	2575.00	RR	LRU
57	2	4	16.77672	1130.00	RR	LRU
58	0	1	1.32732	886.00	RR	LRU
59	1	5	2.45244	320.30	RR	LRU
60	0	1	0.72672	1300.00	RR	LRU
61	0	1	0.96897	1300.00	RR	LRU
62	1	1	1.07507	456.90	RR	LRU
63	1	1	0.75475	309.70	RR	LRU
64	1	1	3.56756	945.50	RR	LRU
65	1	1	2.11010	288.00	RR	LRU

66	3	2	19.16312	578.00	RR	LRU
67	2	2	12.61259	708.00	RR	LRU
68	0	1	3.55354	4000.00	RR	LRU
69	0	2	0.18018	412.00	RR	LRU
70	1	2	1.21721	262.10	RR	LRU
71	2	2	14.64261	1358.00	RR	LRU
72	0	1	3.00899	1874.00	RR	LRU
73	1	1	2.85885	280.00	RR	LRU
74	0	1	1.11111	3658.00	RR	LRU
75	1	1	1.66766	335.00	RR	LRU
76	1	2	6.48247	2129.00	RR	LRU
77	0	1	2.60659	2166.00	RR	LRU
78	1	1	2.22621	1158.00	RR	LRU
79	0	1	3.35134	3550.00	RR	LRU
80	1	1	7.57156	3290.00	RR	LRU
81	0	1	0.82082	802.40	RR	LRU
82	1	1	4.42041	1115.00	RR	LRU
83	2	2	4.00398	267.80	RR	LRU
84	2	1	9.46745	1051.00	RR	LRU
85	1	1	2.35434	875.50	RR	LRU
86	1	2	4.34833	786.50	RR	LRU
87	1	1	2.58458	1401.00	RR	LRU
88	0	1	1.09509	1588.00	RR	LRU
89	1	1	14.73270	6180.00	RR	LRU
90	0	1	0.90290	2553.00	RR	LRU
91	0	2	2.03803	2292.00	RR	LRU
92	0	1	0.98899	831.20	RR	LRU
93	1	1	2.02602	641.69	RR	LRU
94	0	1	2.00198	1421.00	RR	LRU
95	1	1	3.32332	1656.00	RR	LRU
96	1	1	2.36235	165.00	RR	LRU
97	0	1	1.21921	1656.00	RR	LRU
98	1	1	1.10710	591.20	RR	LRU
99	0	1	11.56753	21858.00	RR	LRU
100	0	2	2.11410	1275.00	RR	LRU
101	0	1	1.53553	2386.00	RR	LRU
102	1	1	0.65666	253.60	RR	LRU
103	2	1	8.45844	605.64	RR	LRU
104	1	1	3.77575	1308.00	RR	LRU
105	2	2	7.27926	457.30	RR	LRU
106	1	1	4.73071	1224.00	RR	LRU
107	0	1	1.86586	1359.00	RR	LRU
108	4	1	9.28536	4000.00	RRR	LRU
109	1	1	3.64124	2560.00	RRR	LRU
110	0	1	4.82480	12389.00	RR	LRU
111	1	2	1.39739	430.50	RR	LRU
112	0	1	1.11111	1082.00	RR	LRU
113	0	1	1.33733	892.00	RR	LRU
114	0	2	1.04504	1429.00	RR	LRU
115	1	2	2.53893	1400.00	RR	LRU
116	1	2	1.06506	250.00	RR	LRU
117	1	1	1.76776	471.70	RR	LRU
118	2	1	8.98096	719.70	RR	LRU
119	0	1	1.40940	1686.00	RR	LRU
120	1	1	4.35234	1449.00	RR	LRU
121	1	2	1.59758	570.50	RR	LRU
122	1	1	2.22421	1278.00	RR	LRU
123	1	1	1.54594	886.00	RR	LRU
124	1	3	1.54955	659.40	RR	LRU
125	0	1	1.50349	1119.00	RR	LRU
126	2	2	8.41240	702.60	RR	LRU
127	1	1	17.86976	5954.00	RR	LRU
128	1	1	9.19317	2380.00	RR	LRU
129	1	1	1.56956	650.00	RR	LRU
130	0	1	0.18018	480.00	RR	LRU
131	0	1	1.28328	840.00	RR	LRU

132	1	1	10.92090	5000.00	RR	LRU
133	1	1	1.86566	1010.00	RR	LRU
134	1	1	1.81581	718.75	RR	LRU
135	2	2	4.66865	279.00	RR	LRU
136	1	1	1.34534	129.00	RR	LRU
137	1	1	12.51249	3097.00	RR	LRU
138	1	2	3.53951	696.00	RR	LRU
139	1	1	12.20018	3971.00	RR	LRU
140	1	1	8.40638	2760.00	RR	LRU
141	2	1	18.11806	2711.00	RR	LRU
142	1	1	25.42134	9674.00	RR	LRU
143	0	1	1.41341	3275.00	RR	LRU
144	2	2	15.30728	1615.00	RR	LRU
145	1	1	0.67467	236.10	RR	LRU
146	0	1	0.80280	1674.00	RR	LRU
147	1	2	3.49949	436.00	RR	LRU
148	0	1	20.75862	35511.00	RR	LRU
149	0	1	2.25624	1472.00	RR	LRU
150	0	1	3.62762	2730.00	RR	LRU
151	0	1	17.40334	17694.00	RR	LRU
152	0	1	5.96394	3914.00	RR	LRU
153	0	1	3.59359	3223.00	RR	LRU
154	1	1	3.48947	1183.00	RR	LRU
155	0	1	1.58157	1471.00	RR	LRU
156	2	1	1.69859	1584.00	RRR	LRU
157	2	1	3.56697	41566.00	RRR	LRU
158	1	1	0.23000	41566.00	RRR	LRU
159	3	1	1.74018	3668.00	RRR	LRU
160	2	1	1.01338	1200.00	RRR	LRU
161	1	1	1.73880	13000.00	RRR	LRU
162	1	1	1.18174	13000.00	RRR	LRU
163	17	1	16.33919	51886.00	RRR	LRU
164	6	1	6.78560	39000.00	RRR	LRU
165	8	1	8.50310	40485.00	RRR	LRU
166	9	1	9.07718	40102.00	RRR	LRU
167	4	1	2.91548	5551.00	RRR	LRU
168	7	1	3.87596	2200.00	RRR	LRU
169	1	1	1.12884	5352.00	RRR	LRU
170	3	1	1.91958	4687.00	RRR	LRU
171	11	1	11.02481	45581.00	RRR	LRU
172	1	1	0.57367	29011.00	RRR	LRU
173	6	1	4.60046	15721.00	RRR	LRU
174	5	1	4.38978	11520.00	RRR	LRU
175	5	1	12.59189	32182.00	RRR	LRU
176	1	1	0.29566	566.50	RRR	LRU
177	2	1	0.71001	1200.00	RRR	LRU
178	1	1	2.89770	4496.00	RRR	LRU
179	3	1	2.98540	21265.00	RRR	LRU
180	12	1	11.13084	40606.00	RRR	LRU
181	0	1	1.94994	1715.00	RR	SRU
182	0	1	0.62662	13245.00	RR	SRU
183	0	1	1.42743	13260.00	RR	SRU
184	0	2	1.20120	3539.70	RR	SRU
185	0	1	0.08609	6761.00	RR	SRU
186	0	1	0.57858	5300.00	RR	SRU
187	0	1	0.62662	7300.00	RR	SRU
188	0	1	0.71471	2162.00	RR	SRU
189	0	1	1.00100	2554.00	RR	SRU
190	0	1	0.45846	2100.00	RR	SRU
191	0	1	1.13914	2550.00	RR	SRU
192	0	1	0.79275	1250.00	RR	SRU
193	0	1	4.25224	2500.00	RR	SRU
194	0	1	5.38737	2200.00	RR	SRU
195	0	1	1.13713	839.30	RR	SRU
196	0	1	1.28728	758.00	RR	SRU
197	0	1	0.80080	4166.00	RR	SRU

198	0	1	0.21621	891.90	RR	SRU
199	0	2	3.76374	1246.00	RR	SRU
200	0	1	15.19716	4000.00	RR	SRU
201	0	1	1.01501	800.00	RR	SRU
202	0	2	3.22721	640.00	RR	SRU
203	0	1	3.32931	1535.00	RR	SRU
204	0	1	5.30329	630.60	RR	SRU
205	0	1	1.52151	415.90	RR	SRU
206	0	1	1.43944	547.90	RR	SRU
207	0	1	3.24923	574.00	RR	SRU
208	0	2	4.91689	1174.00	RR	SRU
209	0	1	4.77476	1123.00	RR	SRU
210	0	1	0.92092	526.30	RR	SRU
211	0	1	6.83081	1674.00	RR	SRU
212	0	1	1.68768	434.00	RR	SRU
213	0	1	0.63463	668.50	RR	SRU
214	0	1	1.60760	190.60	RR	SRU
215	0	1	1.30529	345.00	RR	SRU
216	0	1	1.00100	772.50	RR	SRU
217	0	1	2.07005	2450.00	RR	SRU
218	0	1	4.95294	3846.00	RR	SRU
219	0	1	0.55095	3710.00	RR	SRU
220	0	1	2.99497	1654.00	RR	SRU
221	0	1	4.68866	1693.00	RR	SRU
222	0	1	3.56555	2563.00	RR	SRU
223	0	2	0.06807	556.10	RR	SRU
224	0	2	0.13614	685.00	RR	SRU
225	0	1	9.82379	6026.00	RR	SRU
226	0	1	1.36536	1972.00	RR	SRU
227	0	1	10.01398	3250.00	RR	SRU
228	0	1	2.17817	971.90	RR	SRU
229	0	1	2.17817	2208.00	RR	SRU
230	0	1	2.17817	217.60	RR	SRU
231	0	1	2.00198	2052.00	RR	SRU
232	0	1	1.25925	292.50	RR	SRU
233	0	1	3.90389	490.30	RR	SRU
234	0	1	1.80179	321.40	RR	SRU
235	0	1	3.88586	327.50	RR	SRU
236	0	1	2.20219	734.40	RR	SRU
237	0	1	2.55293	322.40	RR	SRU
238	0	1	2.85885	1740.00	RR	SRU
239	0	1	3.00499	854.90	RR	SRU
240	0	1	15.17715	8310.00	RR	SRU
241	0	1	1.80179	348.10	RR	SRU
242	0	1	2.12812	381.10	RR	SRU
243	0	1	0.00399	3033.00	RR	SRU
244	0	1	2.66665	473.80	RR	SRU
245	0	1	1.53953	922.40	RR	SRU
246	0	4	5.00498	690.70	RR	SRU
247	0	1	1.42141	1639.00	RR	SRU
248	0	8	3.63563	372.90	RR	SRU
249	0	15	7.29728	653.00	RR	SRU
250	0	1	2.27026	3114.00	RR	SRU
251	0	1	1.66766	736.70	RR	SRU
252	0	1	0.94094	1484.00	RR	SRU
253	0	1	10.13811	7694.00	RR	SRU
254	0	1	3.65564	794.30	RR	SRU